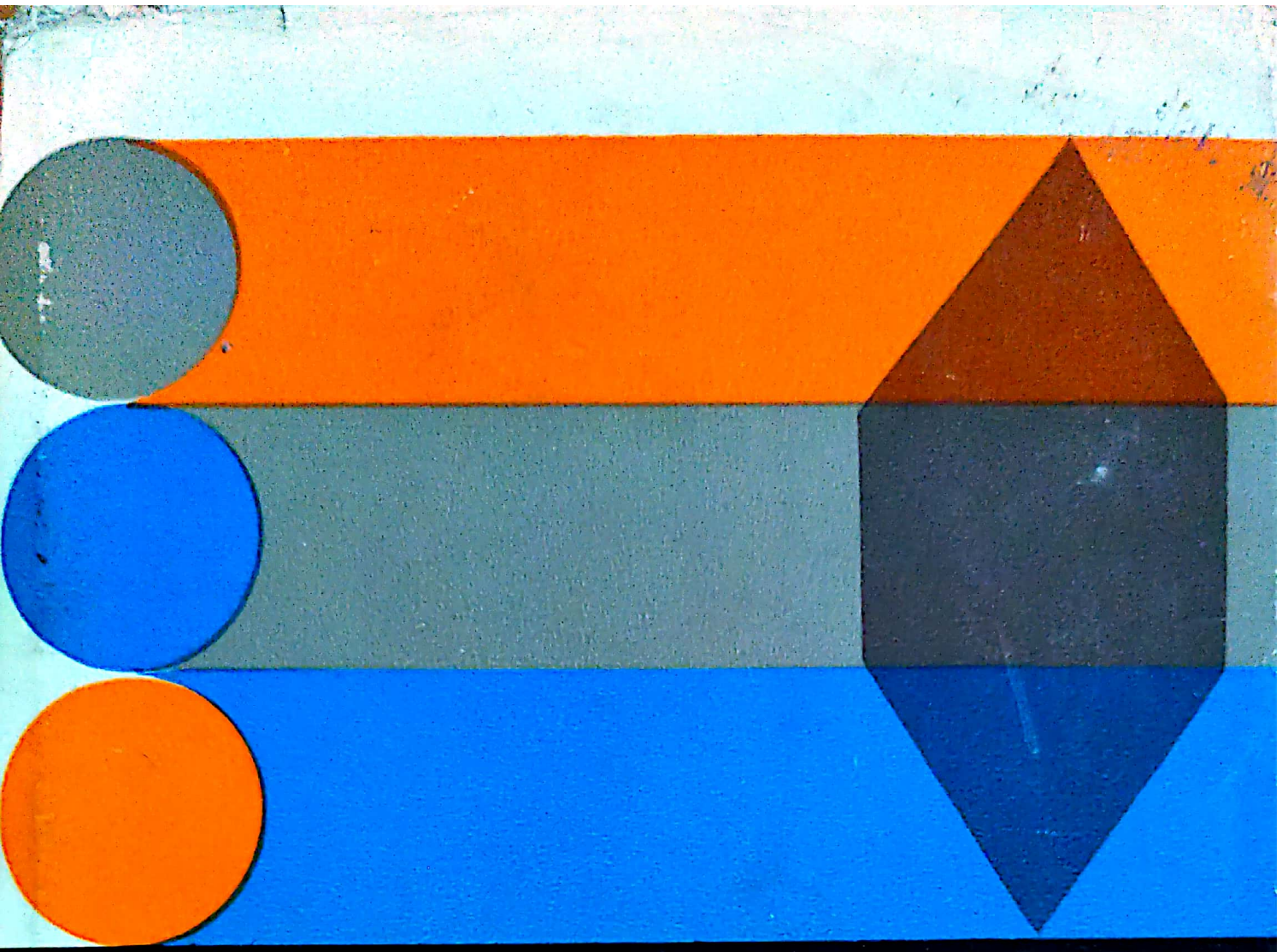


Mircea Gheorghiu
Petru Filip

aplicațiile calculatoarelor în chimie

Scanned with OKEN Scanner

APLICAȚIILE CALCULATOARELOR IN CHIMIE

1000240808101175

Editura Științifică

Chimia este o știință experimentală și ca atare manipulează o cantitate enormă de date. Înțelegerea fenomenelor chimice este, printre altele, dependentă și de modul în care este realizată prelucrarea acestor date. Chimia a primit în ultimii 30 de ani o infuzie importantă de matematică aplicată și de fizică, iar în ultima vreme beneficiază de un mijloc extraordinar de eficient de prelucrare a datelor: calculatoarele.

Începînd din anii '60, calculatoarele au început să fie utilizate intensiv în știință și tehnică, și, spre marea satisfacție a chimiștilor, chimia este una din marile beneficiare ale erei computerelor.

Mircea Gheorghiu
Petru Filip

Scanned with OKEN Scanner

Mircea Gheorghiu • Petru Filip

L. Ivan 15.09.73

APLICAŢIILE CALCULATOARELOR ÎN CHIMIE

Editura Ştiinţifică • Bucureşti, 1973

Scanned with OKEN Scanner

PREFAŢĂ

Fără îndoială, chimia a fost şi va fi întotdeauna o ştiinţă experimentală. Dar, ca orice ramură a ştiinţei zilelor noastre, ea beneficiază de metodele matematicii şi fizicii. Aplicaţiile calculului numeric la probleme de chimie sînt foarte diverse, referindu-se la metode statistice de diverse nivele de sofisticare, integrarea ecuaţiilor diferenţiale de grade de dificultate extrem de variate, probleme de valori proprii etc.

De multe ori o prelucrare inteligentă a datelor experimentale are drept consecinţă o mai bună înţelegere a fenomenologiei chimice ca efect al ordonării şi sistematizării faptelor experimentale. Chimia este o ştiinţă aplicată, dar asta nu înseamnă automat că orice substanţă nouă descoperită va avea aplicaţii practice imediate, fie ca un medicament valoros, fie ca un polimer sintetic cu proprietăţi deosebite. Dar, înainte de toate ar trebui să înţelegem ce anume condiţie sau condiţii structurale determină ca un compus să aibă o anumită aplicabilitate. În anumite stadii ale cercetării este suficient să cunoaştem numai o parte din elementele care determină anumite proprietăţi. Oare cîte din substanţele deja cunoscute au proprietăţi pe care o întreagă omenire le doreşte de zeci de ani, sau care ar trebui să fie structura unui compus care să posede, spre exemplu, o acţiune anticanceroasă? Desigur în cele din urmă vom cunoaşte şi aceste condiţii!

Dar problemele care ne frămîntă nu se rezolvă imediat. Ori cît de simple sau de complicate ar fi, rezolvarea lor necesită eforturi deosebite. Care ar fi procedeele, tehnicile experimentale, încotro trebuie să ne dirijăm eforturile, ce fel de experimente trebuie să conducem pentru a realiza scopul propus? Iată cîteva întrebări de importanţă vitală. Nu există o reţetă standard. Succesul implică în primul rînd cunoştinţe temeinice atît din

Scanned with OKEN Scanner

profesiunea de bază cît şi din ştiinţele limitrofe, între care anumite aspecte de matematică deţin un loc prioritar. La aceasta se adaugă întotdeauna şi ceva inspiraţie şi fantezie.

În ultimii 20 de ani chimia a cunoscut o evoluţie exponenţială, dar toate acestea se datoresc în primul rînd faptului că înţelegem mai bine legile ei. Aceste legi ne permit, de exemplu, să realizăm sinteze, cu un anumit curs steric, deoarece numai compusul care posedă o anumită aranjare sterică este, de exemplu, biologic activ, sau posedă proprietăţile dorite numai acel polimer în care există o anumită aranjare spaţială a monomerilor.

Fără îndoială că, cu cît cunoaştem şi înţelegem mai bine fenomenologia proceselor chimice şi structura compuşilor chimici, cu atît progresul chimiei va fi mai mare.

O parte din această cunoaştere şi înţelegere o datorăm şi calculatoarelor. Dar, existenţa calculatoarelor nu înseamnă automat şi progres. Noi trebuie să ştim să le utilizăm în conformitate cu scopul nostru, în aşa fel încît să constituie un auxiliar preţios în prelucrarea informaţiilor chimice de natură pur experimentală, fie să ne ajute să înţelegem anumite date spectrale în vederea stabilirii structurii compuşilor chimici, fie să ne călăuzească spre acele experienţe care într-adevăr au semnificaţie şi să înlăturăm acele experimente care nu prezintă interes. Desigur că, în vederea concretizării unui scop, trebuie să cunoaştem în primul rînd ce tehnici de lucru rezolvă problematica noastră.

Această carte reprezintă un răspuns parţial la problematica formulată mai înainte, răspuns în limitele concepţiei despre acest subiect a doi chimişti organicieni, de educaţie experimentală, care au utilizat de-a lungul anilor calculatorul pentru rezolvarea unora din problemele întîlnite în practica lor.

Am spus *parţial*, pentru că în primul rînd problematica aplicaţiilor calculatoarelor în chimie este foarte vastă şi în consecinţă ea nu mai poate fi o experienţă a unu, doi sau chiar a mai mulţi chimişti, în al doilea rînd, materialul acestei cărţi reflectă oarecum preocupările autorilor. Din acest motiv nu am inclus utilizarea calculatoarelor pentru determinări de structuri cristaline cu raze X, determinări de structuri cu difracţie electro-

Scanned with OKEN Scanner

nică, simulări de spectre IR, RES. În al treilea rînd, chiar subiectele tratate sînt prezentate într-o formă simplificată; astăzi fiecare capitol, dacă am considera chiar şi cele mai sofisticate nivele de tratare, poate constitui materialul unei cărţi relativ voluminoase.

O carte de aplicaţie, de genul celei de faţă, trebuie să conţină simultan şi anumite aspecte de calcul numeric. Noi am ţinut cont de acest lucru, în măsura în care prezentarea unor astfel de aspecte nu a perturbat o balanţă calcul numeric/aplicaţii pe calculator, care conform scopului propus trebuie să fie în favoarea ultimei.

Unul din autori (M.G.) datorează pasiunea, inspiraţia şi concretizarea practică în utilizarea calculatoarelor în chimie prof. Roald Hoffmann de la Cornell University, Ithaca, New York.

Regretatul prof. C.D. Neniţescu ne-a sprijinit şi încurajat, încă de la primele noastre încercări, a fost alături de noi tocmai în cele mai dificile şi zbuciumate clipe, clipele începutului.

Dorim să mulţumim conducerilor Centrului de calcul al Universităţii şi Institutului politehnic din Bucureşti, unde am reuşit să concretizăm o bună parte din proiectele noastre.

Totodată ţinem să ne cerem scuze cititorilor pentru eventualele erori care s-au strecurat în text.

AUTORII

Scanned with OKEN Scanner

CUPRINS

CAPITOLUL 1. SUMAR DE LIMBAJ FØRTRAN ŞI PROGRAMARE . . 13

1.1. Generalităţi 13
1.2. Instrucţiuni neexecutabile 15
1.2.1. Declaraţiile de tip 15
1.2.2. Instrucţiunile de alocare a memoriei interne 16
1.2.3. Specificaţiile FØRMAT 19
1.3. Instrucţiuni executabile. 31
1.3.1. Instrucţiuni de intrare-ieşire 31
1.3.2. Instrucţiuni de atribuire. 35
1.3.3. Ruperile de secvenţă — instrucţiuni de salt 41
1.3.4. Instrucţiuni de ciclare. 47
1.4. Subprograme 49
1.4.1. Funcţii 49
1.4.2. Subrutine 52
1.5. Operatorul DATA 54

CAPITOLUL 2. METODE NUMERICE APLICATE 56

2.1. Integrare numerică. 57
2.1.1. Teorie generală. Formule Newton-Cotes 57
2.1.2. Programe 60
2.1.3. Aplicaţii. Variaţia căldurii de reacţie cu temperatura . . . 61
2.2. Rezolvarea ecuaţiilor diferenţiale ordinare 64
2.2.1. Clasificarea metodelor. 64
2.2.2. Metode pas cu pas. 65
2.2.3. Programe. Metoda Heun. Metoda Runge-Kutta 68
2.2.4. Metode cu mai mulţi paşi 71
2.2.5. Programe. Metoda Hamming. 73
2.2.6. Simularea reacţiilor chimice 75
2.3. Metode statistice 79
2.3.1. Definiţii şi notaţii. 81
2.3.2. Parametrii statistici ai unei probe 83
2.3.3. Program pentru calculul parametrilor statistici ai unei probe 84

Scanned with OKEN Scanner

2.3.4. Regresie multiliniară 85
2.3.5. Program pentru regresie multiliniară 88

2.4. Calcul cu matrice. 93
2.4.1. Definiţii şi noţiuni de bază 93
2.4.2. Programe. Operaţii cu matrice pătrate. Tipărirea matricelor 96
2.4.3. Rezolvarea sistemelor de ecuaţii liniare. 99
2.4.4. Programe. Metoda Gauss-Jordan. Metoda Gauss-Siedel 101
2.4.5. Probleme de valori proprii 106

2.5. Rezolvarea numerică a ecuaţiilor diferenţiale cu derivate parţiale 120
2.5.1. Utilizarea diferenţelor finite 121

CAPITOLUL 3. APLICAŢII ÎN CHIMIA CUANTICĂ ŞI CINETICA CHIMICĂ 126

3.1. Calculul orbitalilor moleculari. 126
3.1.1. Introducere 126
3.1.2. Aspecte generale ale metodelor de calcul al orbitalilor moleculari. 130
3.1.3. Metoda Hückel de calcul a orbitalilor moleculari . . . 137
3.1.4. Calculul structurii electronice în aproximaţia Pariser-Parr-Pople 158
3.1.5. Metoda Hückel extinsă 185
3.1.6. Metode autocoerente de calcul al structurii electronice a moleculelor folosind toţi electronii de valenţă 207

3.2. Unele aplicaţii în analiza spectrelor de rezonanţă magnetică nucleară. 222

3.3. Calculul coordonatelor nucleelor atomice dintr-o moleculă 237

3.4. Determinări cinetice 246
3.4.1. Cazuri simple 247
3.4.2. Cazuri complexe 254

BIBLIOGRAFIE 261
GLOSAR 264

Scanned with OKEN Scanner

CAPITOLUL 1 SUMAR DE LIMBAJ FØRTRAN ŞI PROGRAMARE

1.1. GENERALITĂŢI

În principiu, cu ajutorul unui calculator electronic se poate rezolva orice tip de problemă numerică, cu condiţia ca modul de rezolvare să poată fi descris în mod univoc iar rezultatul să fie obţinut după un număr finit de etape de calcul. Dacă o problemă merită sau nu să fie rezolvată pe calculator este mai mult o chestiune de economie de timp; vom utiliza maşina numai dacă timpul necesar scrierii şi testării programului este mai scurt decît timpul necesar rezolvării problemei cu mijloace clasice sau dacă calculul cere o precizie deosebită.

Odată hotărîţi să utilizăm maşina electronică, prima operaţie necesară este stabilirea algoritmului de rezolvare, adică a unei însuşiri logice şi finite de instrucţiuni care să permită prelucrarea datelor problemei şi obţinerea şi recunoaşterea rezultatelor.

Să considerăm că dorim să scriem un program care să calculeze punctele de topire ale alcanilor normali şi entropiile de topire cînd sînt date numărul de atomi de carbon (n) şi entalpia de topire (ΔH_t). Dacă nu este dată entalpia de topire, atunci să nu se calculeze entropia.

Pentru aceasta vom folosi formula lui Garner

$$T\,(°\mathrm{K}) = \frac{0{,}6085\ n - 1{,}75}{0{,}001491\ n + 0{,}00404} = \frac{\Delta H_t}{\Delta S_t} \qquad (1.1)$$

Această formulă este valabilă numai pentru $n \geqslant 20$.

În aceste condiţii algoritmul ar fi compus din următoarele instrucţiuni:

1. Citeşte n şi ΔH_t.
2. Verifică dacă $n \geqslant 20$. Dacă da, continuă de la instrucţiunea 3. Dacă nu, continuă de la instrucţiunea 8.
3. Calculează T cu formula (1,1).
4. Verifică dacă ΔH_t este diferit de zero. Dacă da, continuă cu instrucţiunea 5; dacă nu, continuă cu instrucţiunea 7.
5. Calculează $\Delta S_t = \Delta H_t / T$.
6. Scrie T, ΔS_t şi apoi opreşte.
7. Scrie T şi apoi opreşte.
8. Scrie $n < 20$ şi apoi opreşte.

Scanned with OKEN Scanner

A doua etapă constă în alcătuirea organigramei programului. Organigrama este de fapt o reprezentare grafică a algoritmului, reprezentare ce uzează de o serie de semne convenționale și care permite eventuale reorganizări ale programului în scopul simplificării sale.

Dacă reluăm exemplul dat înainte vom obține pentru el organigrama din fig. 1.1.

La ora actuală nu pare să existe un set unic de astfel de simboluri, existînd diferențe de la autor la autor.

A treia etapă constă în traducerea organigramei în programul propriu-zis, într-un limbaj de programare oarecare, dar pe care să îl „înțeleagă" mașina de care dispunem. Limbajul la care ne vom referi de aici înainte va fi limbajul FØRTRAN (*FØRmula TRANslation*), limbaj ce se pretează foarte bine la calcule tehnice și științifice.

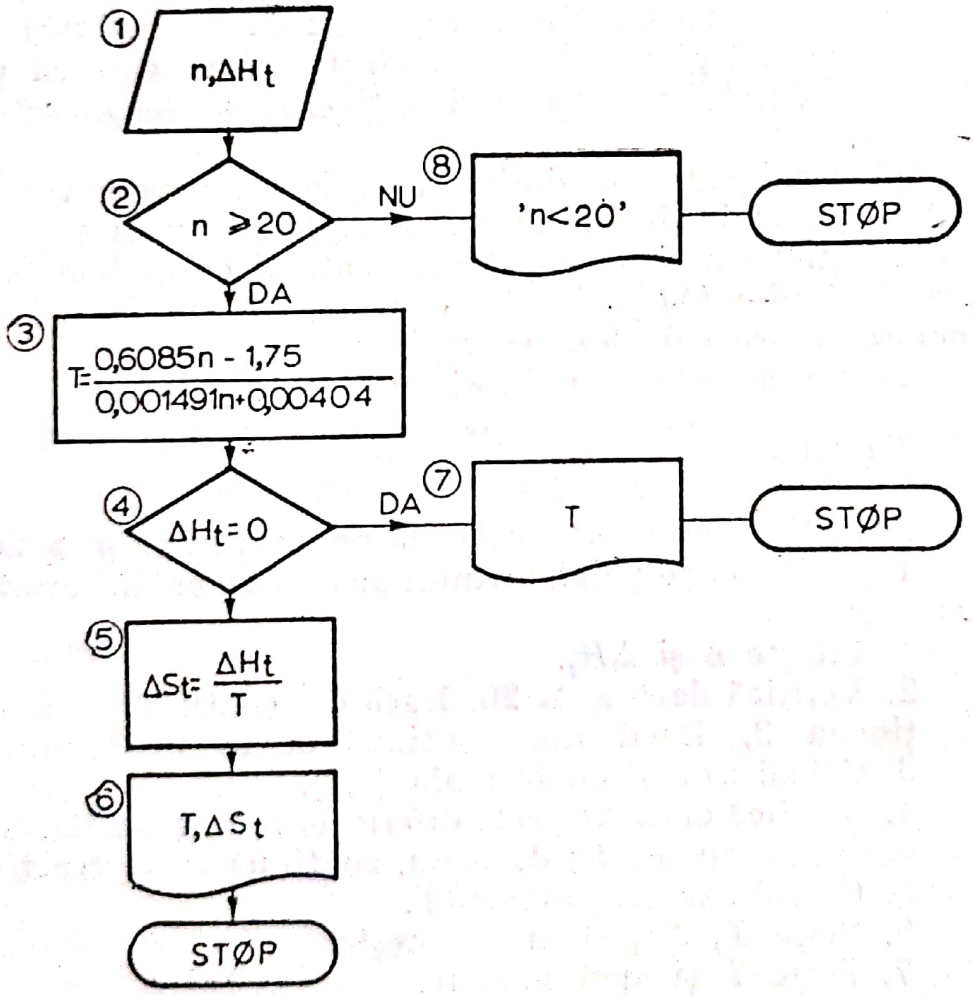


Fig. 1.1.

Scanned with OKEN Scanner

Un program scris în acest limbaj este alcătuit din două tipuri de instrucţiuni: instrucţiuni care servesc efectuării calculelor propriu-zise şi care din acest motiv se numesc instrucţiuni executabile, precum şi o serie de instrucţiuni care precizează în ce fel vor fi tratate de calculator anumite mărimi necesare în calcule, sub ce formă vor fi tipărite rezultatele sau cum se va organiza memoria calculatorului. Acest al doilea tip de instrucţiuni se numesc instrucţiuni neexecutabile. Dintre ele fac parte:

— declaraţiile de tip;
— instrucţiunile de alocare a memoriei;
— specificaţiile FØRMAT,

Instrucţiunile executabile sînt:

— instrucţiunile de atribuire;
— ruperile de secvenţă;
— instrucţiunile de ciclare;
— instrucţiunile de introducere şi extragerea de date.

1.2. INSTRUCŢIUNI NEEXECUTABILE

1.2.1. DECLARAŢIILE DE TIP

În cadrul limbajului FØRTRAN se poate opera cu patru tipuri de mărimi: reale, întregi, complexe şi logice. Primele trei pot exista în două variante de precizie: simplă şi dublă. În cazul preciziei simple mărimile apar cu maximum şase cifre semnificative, iar în cazul preciziei duble cu douăsprezece. Aceste valori pot diferi de la o marcă de calculator la alta.

Declaraţiile de tip sînt instrucţiuni neexecutabile, deoarece nu reprezintă o etapă de calcul, ci doar specifică tipul mărimilor cu care va opera programul. Ele sînt primele instrucţiuni care trebuie să apară în program. În cazul în care programul nu conţine declaraţii de tip este luată în consideraţie următoarea convenţie:

1. Toate mărimile sînt în simplă precizie.
2. Toate variabilele şi constantele ale căror nume începe cu literele A—H, Ø—Z sînt reale.
3. Toate variabilele şi constantele ale căror nume începe cu literele I, J, K, L, M, N sînt întregi.
4. În program nu apar variabile logice sau complexe.

Pentru unii s-ar putea ca declaraţiile de tip să pară inutile din moment ce există o convenţie care fixează tipul variabilelor.

Scanned with OKEN Scanner

Aceste instrucţiuni permit însă utilizarea în calcule a unor nume de variabile (nume ce nu corespund convenţiei de bază) care să fie mai uşor de reţinut. De exemplu, este foarte comod să se utilizeze numele de LAMBDA pentru a desemna într-un calcul de schimbător de căldură conductibilitatea termică care are drept simbol aproape general consacrat λ.

Declaraţiile de tip pot fi explicite sau implicite.

Pentru declaraţiile de tip explicite se foloseşte următorul mod de scriere:

INTEGER REAL DØUBLE PRECISIØN CØMPLEX LØGICAL	urmat de numele variabilelor matricilor sau funcţiilor ale căror tip vrem să îl declarăm.

În cazul declaraţiilor de tip implicite facem de fapt o nouă convenţie privind literele cu care încep diversele tipuri de mărimi:

$$\text{IMPLICIT} \left\{\begin{matrix}\text{INTEGER}\\ \text{REAL}\\ \text{CØMPLEX}\\ \text{LØGICAL}\end{matrix}\right\} (l_1, l_2, \ldots), \ldots \left\{\begin{matrix}\text{INTEGER}\\ \text{REAL}\\ \text{CØMPLEX}\\ \text{LØGICAL}\end{matrix}\right\} (l_m, l_n, \ldots), \ldots$$

De exemplu:

```
IMPLICIT REAL (J,L—N), INTEGER (A,C,P—T)
```

Această nouă convenţie are prioritate faţă de convenţia standard în regiunile unde ele se suprapun. Astfel, în exemplul dat mai sus, cu J încep nume de mărimi reale dar K—N rămîn iniţiale pentru mărimi întregi. Aceste moduri de scriere nu sînt unice, putînd exista diferenţe de la o marcă de calculator la alta. În general, tot limbajul FØRTRAN prezintă mici diferenţe de la o maşină la alta. Acest lucru face necesară consultarea ghidului de programare al maşinii pe care se va lucra efectiv.

1.2.2. INSTRUCŢIUNILE DE ALOCARE A MEMORIEI INTERNE

Există trei asemenea instrucţiuni:

```
DIMENSIØN
EQUIVALENCE
CØMMØN
```


Scanned with OKEN Scanner

Dacă prima dintre ele servește exclusiv rezervării de memorie pentru înmagazinarea unor matrici sau a unei serii de valori ale unei variabile, ultimele două sînt utilizate mai ales pentru economisirea memoriei, instrucțiunea EQUIVALENCE permițînd stocarea unei matrici în spațiul rezervat alteia ale cărei valori nu mai sînt necesare, iar instrucțiunea CØMMØN permite rezervarea de memorie comună mai multor segmente de program într-un spațiu unic.

Instrucțiunea DIMENSIØN are următoarea formă:

DIMENSIØN A(i_1, i_2,...), B(j_1,j_2,...),...

unde: B, A reprezintă nume de matrice sau variabilă;
j_k, i_k — valoarea maximă a indicelui respectiv.
De exemplu:

```
DIMENSIØN ALFA (3,7), GAMA (12)
```

Numărul maxim de indici posibili variază de la un tip de mașină la altul.

La unele dintre mașini, în cazul subprogramelor se poate lăsa dimensiunea maximă nespecificată; în acest caz, dimensiunea de lucru trebuie să fie transmisă prin secvența de chemare a subrutinei.

De exemplu:

```
SUBRØUTINE ØMEGA (A1, X, NMAX, MMAX,Y)
DIMENSIØN A1 (NMAX), X (MMAX), Y(NMAX, MMAX)
```

Instrucțiunea EQUIVALENCE are următoarea formă:

EQUIVALENCE (listă$_1$), (listă$_2$),...

unde o listă conține variabile neindiciate sau variabile indiciate prin constante, separate prin virgule. Efectul este acela de a da tuturor variabilelor din listă aceeași adresă în memorie. Acest efect este tranzitiv și, ca urmare, dacă elementele listei fac parte din matrici, începînd cu termenii din listă se suprapun și adresele celorlalte elemente de matrice.

De exemplu:

```
DIMENSIØN C(10), F(3,4)
EQUIVALENCE (A,C(4), F(3,3)), (P,J,C(9))
```


Scanned with OKEN Scanner

are următorul efect:

```
                                        F(1,1)
                                        F(2,1)
                                        F(3,1)
                                        F(1,2)
                                        F(2,2)
C( 1). . . . . .F(3,2)
C( 2)                                   F(1,3)
C( 3)                                   F(2,3)
C( 4). . . .A. . . . . . . . . . .F(3,3)
C( 5)                                   F(1,4)
C( 6)                                   F(2,4)
C( 7)                                   F(3,4)
C( 8)
C( 9). . . .P. . . .J. . . . . . . . . .
C(10)
```

Datorită acestei tranzitivităţi trebuie acordată o atenţie deosebită în cazul utilizării acestei instrucţiuni.

Instrucţiunea CØMMØN are forma identică cu instrucţiunea DIMENSIØN, substituind-o chiar pentru mărimile pentru care se alocă memorie prin ea.

De exemplu:

```
DIMENSIØN A(30), B(2,9), C(3,4,8)
CØMMØN    A,B
```

este echivalent cu

```
DIMENSIØN C(3,4,8)
CØMMØN    A(30), B(2,9)
```

Instrucţiunea CØMMØN mai poate apărea în altă formă, şi anume, în cazul în care este nevoie de mai multe blocuri comune se foloseşte blocul comun cu etichetă. În acest caz, instrucţiunea are următoarea formă:

CØMMØN/etichetă$_1$/ listă$_1$/ etichetă$_2$/ listă$_2$ . . . . . .

În acest caz fiecare secţiune a programului va avea acces numai la blocurile de care are nevoie.

De exemplu:

— programul principal

```
CØMMØN /ET1/A(68),B(4,5)/ET2/X(5) /ET3/C(35), P(12,12)
```


Scanned with OKEN Scanner

— subrutina nr. 1

```
CØMMØN /ET1/ A(8), B(4,5)
```

— subrutina nr. 2

```
CØMMØN /ET3/ C(35), P(12,12) /ET2/ X(5)
```

— subrutina nr. 3

```
CØMMØN /ET3/ C(5,7), P(144)
```

Ca urmare a modului de scriere din exemplu, în cazul celei de a treia subrutine, conţinutul vectorului C este transferat sub forma unei matrici, iar matricea pătrată P se transformă într-un vector. Tot aşa de bine întreg conţinutul oricăruia dintre blocurile comune se poate transforma în valori ale unei singure variabile dacă noii variabile i se prevede o dimensiune egală cu a întregului bloc. Astfel comunul ET1 ar putea deveni într-o subrutină oarecare CØMMØN/ET1/Z(28), unde primele opt valori ar fi cele care au fost ale vectorului A, iar următoarele douăzeci cele ale matricii B citite pe coloane.

1.2.3. SPECIFICAŢIILE FØRMAT

Toate calculatoarele ştiinţifice au reprezentarea internă a numerelor făcută în sistem binar. Aceasta face necesară conversia datelor din sistem zecimal în binar şi invers, ori de cîte ori se face o operaţie de introducere sau extragere de date din maşină. Instrucţiunea FØRMAT este cea care permite conversia automată în ambele sensuri, funcţie de tipul variabilelor cu care se lucrează (întregi, reale, logice etc.). Această instrucţiune permite de asemenea decupajul informaţiei, adică specificarea gabaritului (număr total de cifre, poziţia virgulei zecimale) fiecărei mărimi numerice introduse sau extrase.

O instrucţiune format se poate plasa oriunde în program, însă din motive de comoditate este bine ca toate instrucţiunile de acest fel să fie plasate la începutul programului, după instrucţiunile de alocare a memoriei. În felul acesta, este mult mai uşor de găsit cartela cu instrucţiunea format pe care eventual dorim să o modificăm şi, în acelaşi timp, în cazul adăugării unei noi instrucţiuni de format, se poate alege pentru ea mai uşor o etichetă care să nu se repete. În acest sens, este comod ca etichetele să aibă valori grupate. De exemplu, toate etichetele de instrucţiuni format care se referă la operaţii de introducere

Scanned with OKEN Scanner

de date să fie cuprinse între 1000 și 2000, iar cele care se referă la operații de tipărire a rezultatelor să fie cuprinse între 2000 și 3000.

Un ordin FØRMAT are următoarea structură:

m FØRMAT (listă de specificații care precizează conversiile și delimitările ce urmează a se efectua).

Eticheta *m* servește pentru identificarea formatului, în cadrul instrucțiunilor de intrare/ieșire.

Există trei tipuri de specificații:

1. Specificații de punere în pagină.
   X — inserție de spații albe (ieșire); salt de coloană (intrare);
   H — creare de titluri; este echivalent cu scrierea titlului corespunzător între ghilimele (la ieșire);
   / — salt de linie (ieșire); citirea în continuare de pe cartela următoare (intrare);
   T — poziționare;
   — caractere de salt.

2. Specificațiile de tratare a variabilelor:
   I — numere întregi;
   E — numere reale în simplă precizie exprimate în formă exponențială (cu o mantisă și un exponent);
   F — numere reale în simplă precizie exprimate cu o parte întreagă și una fracționară;
   D — numere reale în dublă precizie exprimate în formă exponențială;
   L — variabile logice;
   A — intrări-ieșiri ale caracterelor alfa-numerice.

3. Specificația de încadrare: P.

Aceste specificații (exceptînd / și caracterele de salt) se utilizează însoțite de un „cuantificator", al cărui rol este de a specifica numărul de coloane de cartelă sau de imprimantă care sînt utilizate de specificația respectivă.

De exemplu:

15 X — înseamnă 15 coloane tratate conform specificației X.

1.2.3.1. Explorarea unui FØRMAT

Explorarea unui FØRMAT este făcută simultan cu cea a listei operației de intrare-ieșire (vezi paragraful 1.3.1. pag. 31) care se referă la formatul în cauză. În principiu, acesta trebuie să

Scanned with OKEN Scanner

conţină tot atîtea specificaţii de tipul 2 cîte variabile conţine lista. Dacă această condiţie nu este îndeplinită, pot exista următoarele situaţii :

— dacă FØRMAT-ul este prea lung, este abandonat cînd se termină lista. Dacă după ultima specificaţie de tip 2 utilizată apar una sau mai multe specificaţii de tipul 1, acestea sînt luate în consideraţie :

— dacă FØRMAT-ul este prea scurt el este reexplorat. Aceasta are următoarele efecte :

la intrare — citirea cartelei următoare ;
la ieşire — imprimarea în continuare de la începutul liniei următoare.

Este foarte important de ştiut în ce loc se va produce reexplorarea FØRMAT-ului. Pentru aceasta, trebuie cunoscute proprietăţile factorilor de repetiţie, problemă pe care o vom prezenta ceva mai încolo.

1.2.3.2. Proprietăţile specificaţiilor de tipul 1

O cartelă de date conţine 80 de coloane ; pe fiecare din acestea se poate înscrie o literă, cifră sau un semn special (+, — , /,* etc.). O imprimantă rapidă poate imprima, funcţie de tip, 128 pînă la 160 de semne pe o linie.

Specificaţia X

Forma de utilizare a acestei specificaţii este
*n*X unde *n* este o constantă pozitivă.

Ea are următorul efect :
la intrare : se neglijează conţinutul a *n* coloane;
la ieşire : se inserează *n* spaţii între două regiuni imprimate de pe aceeaşi linie.

Specificaţia H

Se foloseşte sub forma :

*n*H----------

n caractere,
inclusiv spaţiile neimprimate.

Această specificaţie se utilizează în mod special la ieşirea pe imprimantă pentru a pune titluri sau pentru a intercala semne într-o suită de valori numerice.

Scanned with OKEN Scanner

De exemplu :

FØRMAT (1H1,55X, 17H CINETICA CHIMICA///)

Specificaţia / (*linie oblică*)

Această specificaţie are următoarele proprietăţi :

la intrare :

Se întrerupe citirea cartelei şi se trece la citirea cartelei următoare, începînd cu prima coloană.

De exemplu :

FØRMAT (...../ / / ...../)

va avea ca efect ignorarea conţinutului a două cartele şi continuarea citirii de pe a treia.

la ieşire :

Se întrerupe imprimarea pe o linie şi se continuă din prima coloană a liniei următoare. FØRMAT-ul din exemplul de mai sus ar avea ca efect imprimarea după un salt de două linii, pe cea de a treia linie.

Dacă cele trei linii oblice s-ar afla la începutul sau la sfîrşitul formatului, imprimarea în cauză ar fi separată de cea anterioară sau de cea următoare printr-un spaţiu de trei linii. Aceasta se datoreşte faptului că orice început de scriere provoacă trecerea la o nouă linie.

Specificaţia T

Se utilizează sub următoarea formă :

T*n* unde *n* este un număr întreg pozitiv.

Ea are ca proprietăţi :

la intrare :

Instrucţiunea arată că semnele corespunzătoare specificaţiei ce urmează după T*n* încep din coloana a *n*-a a cartelei ;

la ieşire :

Această specificaţie indică coloana $n-1$ drept coloana din care începe imprimarea semnelor corespunzătoare specificaţiei ce urmează după T*n*.

Caracterele de salt

Cu ajutorul acestor specificaţii se poate comanda mişcarea pe verticală a hîrtiei în imprimantă.

Caracterele de salt se utilizează pentru a aranja estetic rezultatele extrase din măsura de calcul. Această specificaţie poate avea una din formele :

1H*s*

'*s*',

Scanned with OKEN Scanner

unde *s* este unul din următoarele semne:

un spațiu neimprimat sau imprimat cu orice semn, exceptînd celelalte caractere de salt: are ca efect trecerea normală la un început de linie;

\+ — provoacă suprimarea trecerii la o nouă linie și supraimprimarea pe linia precedentă;

0 — provoacă un salt de o linie;

1 — provoacă un salt de pagină.

În general este interpretat drept caracter de salt orice semn ce ar urma să fie imprimat pe prima coloană a imprimantei. Din acest motiv este preferabil ca o imprimare să nu fie începută niciodată din prima coloană.

Ca exemple de utilizare a specificațiilor de tipul 1 vom prezenta formate pentru scrierea unui titlu: CINETICA CHIMICA. Dorim ca acest titlu să fie separat printr-o pagină de imprimările anterioare, iar imprimarea următoare lui să fie cu 3 rînduri mai jos.

Titlul să fie plasat relativ la centrul liniei.

Astfel de formate ar fi:

```
FØRMAT(1H1, 55X,9H CINETICA, T68, 'CHIMICA' ///)
FØRMAT(1H1, 55X, 17H CINETICA CHIMICA, ///)
FØRMAT(1H1, T56, 17H CINETICA CHIMICA///)
FØRMAT(1H1, T68, 7HCHIMICA, T56,'∧CINETICA' ///)
```

După cum se vede, specificația H poate fi înlocuită prin scrierea textului corespunzător între ghilimele. De asemenea, se vede că, în cazul specificației T, nu trebuie respectată ordinea crescătoare a numărului coloanelor.

Prin ∧ vom înțelege un spațiu neimprimat.

1.2.3.3. Proprietățile specificațiilor de tipul 2

Specificația I (INTEGER)

Servește la introducerea și extragerea de numere întregi. Forma de utilizare este:

I*n*

unde *n* este numărul de coloane pe cartelă sau imprimată alocate pentru numărul respectiv.

Această specificație are următoarele proprietăți:

Scanned with OKEN Scanner

la ieşire :

Numărul este plasat la dreapta intervalului de n coloane. Semnul, dacă este negativ, se imprimă la stînga celei mai din stînga cifre. Dacă intervalul alocat este mai mare decît cel necesar, spaţiile suplimentare sînt lăsate neimprimate la stînga numărului. Dacă spaţiul rezervat este insuficient, toată zona se imprimă cu asteriscuri.

De exemplu, pentru Z = — 45678, utilizînd formatele I4, I5, I6, I7, I8, pe imprimantă vom obţine

```
* * * *
* * * * *
- 4 5 6 7 8
∧ - 4 5 6 7 8
∧ ∧ - 4 5 6 7 8
```

la intrare :

Numărul trebuie înscris pe maximum n coloane, plasat la dreapta, şi cu semnul —, dacă este minus, în primul spaţiu de la stînga primei cifre. Poziţiile eventual libere de la stînga se lasă neperforate sau se perforează zero. Dacă numărul este chiar zero este bine ca întreaga zonă să se perforeze cu zerouri ; dacă zona se lasă neimprimată, la unele calculatoare, se obţine un zero negativ, care poate introduce erori în cazul în care se fac comparaţii cu această valoare.

Singurele caractere care au voie să apară sub specificaţia I sînt cifrele de la 0 la 9, + , — şi spaţiile neimprimate. Orice alt semn va fi interpretat drept eroare.

Să considerăm o cartelă care are perforat pe ea începînd cu coloana 10.

```
0237 - 56700* ∧ ∧ 4967
```

iar lista asociată instrucţiunii de citire conţine variabilele K1, K2, K3, K4, K5 :

— dacă formatul de citire este 9X, I1, I3, I4, 3X, I3, I3 se va obţine

K1 = 0 K2 = 237 K3 = — 567
K4 = 4 K5 = 967

— dacă citim în format 9X, I2, I2, I6, IX, I4, I2

K1 = 2 K2 = 37 K3 = — 56700
K4 = 49 K5 = 67

Scanned with OKEN Scanner

— dacă citim în forma I9, I5, I6, I3, I3 am obține mesaj de eroare, deoarece în spațiul rezervat pentru K2 apare un — într-un loc nepermis, iar în spațiul lui K3 apare un semn nepermis (*)

Specificația E (EXPØNENT)

Această specificație servește la introducerea și extragerea de numere reale în simplă precizie exprimate printr-o mantisă și un exponent (putere a lui 10).

Specificația are forma:

E*n.m*

unde *n* este numărul total de poziții ocupate de număr, iar *m* indică cîte zecimale conține mantisa. Proprietățile acestei specificații sînt:

la ieșire:

Numărul va fi imprimat sub forma

$$\pm 0.\underbrace{XXX\ldots X}_{m\text{ spații}}E\pm YY$$

în total *n* spații.

Între *m* și *n* trebuie să existe relația

$n \geqslant m + 7$

Cînd semnele sînt pozitive, în locul lor se lasă spații neimprimate.

Valoarea lui *m* nu trebuie să depășească numărul de cifre semnificative cu care lucrează calculatorul, deoarece cifrele suplimentare nu au semnificație fizică.

Indiferent de valoarea lui *m*, ultima cifră a mantisei este obținută prin rotunjire.

Să considerăm de exemplu numărul X = —7.24159.

În formatele E9.2, E11.4, E12.5, E12.6, E13.6 și E14.6 se vor obține următoarele reprezentări:

```
-0.72E 01
-0.7242E 01
-0.72416E 01
**********
-0.724159E 01
^-0.724159E 01
```

După cum se vede din exemplu, dacă nu este satisfăcută relația dintre *m* și *n*, tot spațiul rezervat numărului este imprimat cu asteriscuri.

Scanned with OKEN Scanner

la intrare:

Acest tip de specificaţie permite o deosebită libertate de scriere la intrare. Practic, cu toate că forma generală este:

$$\underbrace{\pm X \ldots XX.\underbrace{YY \ldots\ldots Y}_{m \text{ spaţii}} E \pm ZZ}_{\text{total } n \text{ spaţii}}$$

se obţine în general un rezultat corect, indiferent de modul de scriere, cu condiţia de a utiliza numai caractere permise, şi anume: cifrele de la 0 la 9, semnele +, −, litere E şi spaţiile neimprimate.

Astfel, pentru reprezentarea numărului −7.2415, în formatul E12.5, cele 12 coloane de pe cartelă vor putea avea următorul conţinut:

```
−0.7245E+01
∧−7.245E 00
∧∧−7245E+01
∧∧∧−7245E01
∧∧∧∧−7245E1
∧∧∧∧∧−7.245
```

Regulile practice de utilizare sînt următoarele:

1. Punctul zecimal poate fi perforat oriunde în cadrul mantisei. În acest caz, el are prioritate asupra indicaţiei din format în care *m* nu mai are semnificaţie. Dacă punctul zecimal nu este perforat, poziţia sa este fixată de specificaţia din instrucţiunea FØRMAT. Această regulă este exemplificată în primele trei moduri de scriere din exemplul anterior.

2. Exponentul este separat de mantisă prin E şi semnul +, − sau un spaţiu neimprimat (∧). Exponentul poate să fie nul sau chiar absent. În acest caz, valoarea numărului este egală cu cea a mantisei.

3. Cifrele şi semnele utilizate conform regulilor se plasează la dreapta zonei de *n* coloane. Semnul se scrie la stînga primei cifre din mantisă. La stînga semnului mantisei nu au voie să existe decît spaţii neimprimate.

Specificaţia F (FRACTIØNARY)

Este utilizată pentru introducerea sau extragerea de numere reale exprimate printr-o parte întreagă şi una fracţionară. Numerele sînt în simplă precizie. Forma utilizată este

F*n.m*

Scanned with OKEN Scanner

unde n este numărul total de spaţii ocupate de număr, iar m numărul de zecimale.

la ieşire:

Numărul va apărea reprezentat sub forma

s XX ... X . YY ... Y

unde:

XX...X: parte întreagă,
YY ... Y: parte fracţionară;

s: semnul; dacă este — se înscrie; dacă este + se lasă un spaţiu alb

Intre m şi n trebuie să existe relaţia:

$$n \geqslant m + 2.$$

Dacă vom nota cu p numărul de spaţii ocupate de partea întreagă a numărului,

$$n = m + p + 2.$$

Valoarea lui $(m + p)$ nu are rost să fie mai mare decît numărul de cifre semnificative cu care lucrează calculatorul.

Această specificaţie trebuie folosită numai dacă sîntem siguri dinainte de valoarea pe care o poate avea p. Dacă numărul de întregi depăşeşte spaţiul prevăzut, toată regiunea de n coloane se imprimă cu asteriscuri.

De exemplu, pentru —7.4279 vom obţine în formatele F5.4, F7.5, F7.4, F9.4:

```
* * * * *
* * * * * * *
— 7 . 4 2 7 9
∧ —7 . 4 2 7 9
```

Dacă m este nul, se imprimă numai partea întreagă a numărului. În acest caz, specificaţia trebuie să conţină în mod obligator zeroul. Fn.0 este corect, iar Fn. nu este corect.

Dacă spaţiul rezervat zecimalelor este insuficient, acestea se trunchează prin rotunjire.

la intrare:

Dacă se perforează şi punctul zecimal, numărul poate fi plasat oriunde în interiorul zonei de n coloane.

Dacă nu se perforează şi punctul zecimal, cifrele trebuie plasate la dreapta intervalului, iar numărul de zecimale este cel indicat prin instrucţiunea FØRMAT.

Scanned with OKEN Scanner

Să considerăm o cartelă ce conţine începînd din prima coloană :

24.275 ∧∧∧∧ 12345678.91∧∧12345

Lista de intrare este formată din variabilele A, B, C, D. Pentru formatul F6.3, F10.4, F7.3, F7.0 obţinem

A = 24.275
B = 12.3456
C = 78.9100
D = 1234500.

Pentru formatul F4.1, F6.5, F11.2, F9.5 obţinem

A = 24.2
B = 7.5
C = 12345678.91
D = 12.345

Specificaţia D (DØUBLE)

Această specificaţie se utilizează în mod exclusiv pentru variabilele exprimate în dublă precizie cu o mantisă şi un exponent. Forma de utilizare este :

D*n*.*m*

unde *n* este numărul total de poziţii ocupate, iar *m* — numărul de zecimale al mantisei.

Mantisa are în general valoarea cuprinsă între 0 şi 1.

Tot ce se aplică specificaţiei E este valabil şi pentru specificaţia D, cu următoarele modificări :

1. Numărul maxim de cifre semnificative este dublu ;
2. În loc de E, la imprimantă sau pe cartele, va apărea D.

Specificaţia L (LØGICAL)

Această specificaţie se utilizează exclusiv pentru variabilele logice. Forma ei este :

L*n*

unde *n* este numărul total de spaţii (de obicei $n = 1$).

Proprietăţile acestei specificaţii sînt :

la ieşire :

Dacă variabila are valoarea ADEVARAT, se va imprima un T (*true*) la dreapta zonei, precedat de $n-1$ spaţii neimprimate. Dacă valoarea este FALS, se va imprima un F (*false*) în aceleaşi condiţii ca mai sus.

Scanned with OKEN Scanner

la intrare!

Singurele caractere care sînt recunoscute ca valabile sînt F, T şi spaţiile neimprimate, care sînt interpretate ca F.

Specificaţia A (ALPHANUMERIC)

Această specificaţie serveşte manipulării de text format, din litere, cifre şi semne speciale. Se utilizează pentru a introduce titluri şi comentarii privitoare de obicei la datele de intrare.

Forma utilizată este

A*n*

Valoarea lui *n* este de astădată legată de numărul de caractere care pot fi stocate într-un cuvînt-maşină. La maşinile de tip IBM din seria 360 se utilizează pentru *n* valoarea 4.

Variabilele care utilizează specificaţii A pot fi de orice tip, în acest caz tipul pierzîndu-şi caracterul. Cu aceste variabile nu se pot efectua calcule sau comparaţii.

Utilizarea specificaţiei va reieşi mai bine din exemplele de programe care se dau în celelalte capitole ale cărţii.

Factorii de repetiţie a specificaţiilor

Specificaţiile de tipul 2 pot fi repetate prin scrierea înainte a unui factor de repetiţie întreg, pozitiv, diferit de zero şi egal cu numărul de repetiţii pe care îl dorim:

4F8.3 este echivalent cu F8.3, F8.3, F8.3, F8.3.

Specificaţiile de tip 1 se pot repeta prin acelaşi procedeu, cu deosebirea că specificaţia trebuie scrisă între paranteze:

10(1H*)	este echivalent cu	10H **********
3(/)	este echivalent cu	///

Pentru a repeta o listă de specificaţii de tipuri diferite, lista se închide între paranteze:

2(F7.3,L5/3X,E12.4)
este echivalent cu
F7.3,L5/3X,E12.4,F7.3,L5/3X,E12.4

În cele ce urmează nu ne vom referi niciodată la parantezele exterioare care delimitează lista unei instrucţiuni FORMAT.

Se numeşte GRUP orice listă de specificaţii cuprinse într-o paranteză. Un factor de repetiţie plasat în faţa parantezei din stînga provoacă repetarea grupului.

Scanned with OKEN Scanner

Exemplu:

FØRMAT (3I4,A6/3X,F3.0)

Această listă nu este un grup; următoarea este

FØRMAT (4(I5,10E13.5//))

În general un grup poate fi inclus în alt grup.

Un grup care la rîndul său nu mai este inclus în nici un alt grup se numeşte GRUP COMPLET.

Exemplu:

FØRMAT (10X,3(F6.2,I3/), 2(E12.5,3X,I3))

Grup complet

Şi acum o regulă de importanţă deosebită:

Reexplorarea unui FØRMAT are loc în următoarele condiţii:

— de la începutul acestuia, dacă nu conţine decît specificaţii elementare;

— de la începutul grupului complet, situat cel mai la dreapta.

Vom exemplifica această regulă după ce vom prezenta instrucţiunile de intrare-ieşire.

1.2.3.4. Specificaţia de încadrare P

Se foloseşte asociată cu o specificaţie E, F sau D şi are efectul de a modifica ordinul de mărime al numărului la introducerea sau extragerea din calculator. Forma specificaţiei este:

aP

şi efectele ei sînt:

la intrare: număr intern = număr extern $* 10^{-a}$;

la ieşire: număr extern = număr intern $* 10^{a}$.

La intrare se foloseşte numai împreună cu specificaţia F.

La ieşire, acest factor de încadrare — care are efectul de a deplasa punctul zecimal — cînd se utilizează împreună cu E şi D, nu modifică valoarea numărului, deoarece automat se ajustează şi valoarea exponentului.

1.3. INSTRUCȚIUNI EXECUTABILE

1.3.1. INSTRUCȚIUNI DE INTRARE-IEȘIRE

Pentru ca o operație de introducere sau extragere de date să poată avea loc sînt necesare următoarele informații:

sensul operației (intrare sau ieșire);

natura suportului informației

cartele perforate
bandă magnetică sau disc } intrare

cartele de perforat
hîrtie de imprimat
bandă sau discuri magnetice } ieșire

lista de variabile implicate în operație;

indicații privind modul de reprezentare a mărimilor din listă (prin instrucțiuni FØRMAT).

Forma instrucțiunii de intrare-ieșire depinde în oarecare măsură de suportul cu care are loc comunicarea.

1.3.1.1. I/E pe cartele

Pentru intrare se folosește instrucțiunea:

READ (*m*,*n*), listă de variabile de introdus,

unde *m* reprezintă numărul simbolic al unității periferice (în cazul nostru cititor de cartele)

n — eticheta instrucțiunii FØRMAT care precizează modul de citire al cartelei.

De exemplu:

```
    READ(1,120)I1,RX
120 FØRMAT(I3,2F10,5)
```

Dacă vrem să introducem pe această cale valorile elementelor unei matrici de dimensiune, de exemplu, 3×5, vom scrie:

```
    READ(1,127)((K(I,J), J = 1,5),I = 1,3)
127 FØRMAT (15I4)
```

Se puteau folosi și alte formate. De exemplu:

a) 127 FØRMAT(5I4)

Scanned with OKEN Scanner

în acest caz, pe fiecare cartelă ar fi fost 5 valori şi s-ar fi citit 3 cartele reexplorîndu-se mereu formatul ;

b) 127 FØRMAT(3(5I4/))

în acest caz s-ar fi citit tot de pe trei cartele cîte 5 valori.

Pentru ieşirea pe cartele se foloseşte instrucţiunea :

PUNCH *n*, listă

unde *n* este eticheta formatului.

Această instrucţiune este utilă dacă o serie de date rezultate din calcul vor necesita prelucrări suplimentare, să spunem pe un calculator mai mare ; ele se pot perfora direct în formatul în care vor servi ca date.

1.3.1.2. Ieşirea pe hîrtie de imprimantă

Forma instrucţiunii este :

WRITE (*m*,*n*) listă

unde *m* desemnează de astădată o imprimantă rapidă.

În cazul în care dorim să scriem de exemplu termenii unei matrici 3×5, vom putea utiliza o scriere similară celei de la introducerea matricilor :

WRITE (3,130)((A(I,J), J = 1,5), I = 1,3)

Un exemplu de FØRMAT ar fi :

130 FØRMAT (T10, 'MATRICE' ///, 3(5(3X,E12.5) //)).

Efectul său ar fi inserarea a 3 linii între titlul MATRICE şi prima linie de 5 elemente de matrice. Pe linie, cifrele vor fi separate prin intervale de 3 coloane, iar pe verticală prin două linii.

Deoarece (5 (3X, E12.5)//) formează un grup complet ; în cazul în care el nu ar fi fost repetat prin introducerea factorului de repetiţie, efectul instrucţiunii era acelaşi, căci la reexplorare se lua în considerare numai acest grup.

1.3.1.3. Utilizarea benzilor şi discurilor magnetice

Benzile magnetice se folosesc ca : suport intermediar pentru introducerea şi extragerea informaţiilor şi memorie temporară, de mare capacitate pentru calcule, în care intervin multe rezultate intermediare.

Scanned with OKEN Scanner

Prima utilizare este determinată de faptul că citirea și scrierea pe bandă sînt cam de 100 de ori mai rapide decît citirea de pe cartele, sau, respectiv, scrierea pe imprimantă.

Din acest motiv, informațiile, indiferent de sensul în care circulă, se stochează întîi pe bandă și apoi se transferă în continuare. La acest gen de utilizare datele se citesc de pe bandă ca și cum s-ar citi de pe cartele, sau se scriu pe bandă ca și cum s-ar scrie la imprimantă.

Instrucțiunile corespunzătoare sînt:

READ (m,n) listă
WRITE (l,k) listă

unde m, l sînt numere simbolice ale unor unități de bandă magnetică, iar n, k — etichetele unor instrucțiuni FØRMAT.

Dacă banda este utilizată ca memorie intermediară, nu mai este necesară specificarea formatului, deoarece nu este necesară conversia formei de reprezentare a informațiilor, fiind convenabilă păstrarea formei de reprezentare existentă în memoria operativă.

Forma instrucțiunilor va fi:

READ (m) listă
WRITE (l) listă

unde m, l sînt numere simbolice ce desemnează unități de bandă magnetică.

Aceste instrucțiuni sînt completate cu trei instrucțiuni speciale:

REWIND n — provoacă rebobinarea benzii n pînă aduce sub capul de citire prima înregistrare de pe bandă.

BACKSPACE n — provoacă întoarcerea benzii n la începutul ultimei înregistrări. Astfel, dacă înregistrările se găsesc în formă de „imagini de cartele", se va putea reciti „cartela anterioară".

END FILE n — marchează sfîrșitul unui set de date. Are ca efect imprimarea unui cod special pe banda din unitatea n; codul va fi recunoscut ulterior la citirea benzii.

În cazul utilizării discurilor magnetice (scopul este în special acela de memorie intermediară, cu acces mai rapid decît banda) există patru instrucțiuni:

— DEFINE FILE $n^1(l_1^1, l_2^1, a^1, i^1), n^2(l_1^2, l_2^2, a^2, i^2) \ldots$ Această instrucțiune servește definirii unor fișiere de date. Aici

Scanned with OKEN Scanner

este esențial faptul că înregistrările pe o unitate de discuri sînt de lungime fixă.

Termenii din instrucțiune au următoarele semnificații:

n^i — numărul simbolic al unității de discuri pe care se află fișierul;

l_1^i — numărul de înregistrări din fișier;

l_2^i — numărul de poziții în memorie, necesare pentru fiecare înregistrare;

a^i — o literă (L, U sau E) prin care se determină felul cum are loc schimbul de informații:

L — cu sau fără indicație de format;
U — fără indicație de format;
E — cu indicație de format;

i^i — o variabilă de tip întreg, denumită variabilă asociată; valoarea ei servește indicării înregistrării imediat următoare celei transmise. Valoarea ei se stabilește automat.

Trebuie remarcat că „pozițiile" în memoria operativă definite de l_2 au sensuri diferite în funcție de a.

Dacă a este L sau E, atunci o poziție este egală cu memoria alocată pentru un caracter; pentru specificația U o poziție este egală cu memoria alocată pentru un cuvînt-mașină (la IBM 360, 4 caractere).

Deci dimensiunea efectivă a înregistrării depinde de l_2 și a.

Exemplu:

```
DEFINE FILE 4(500,30,E,K1)
```

pe unitatea 4 se vor stoca 500 de înregistrări de cîte 30 de caractere (*cf* E), care se transmit cu indicație de format.

Formatul corespunzător la care se va referi instrucțiunea de citire sau scriere va trebui să determine analizarea a 30 de poziții; de exemplu (2I5, 4F 5.2).

Accesul direct pentru citire este realizat prin instrucțiunea:

```
READ (n ' e, m) listă
```

unde:

n este numărul simbolic al unității de pe care se citește;

e — o expresie de tip întreg care indică poziția înregistrării ce urmează a fi citită;

m — eticheta instrucțiunii FØRMAT.

Scanned with OKEN Scanner

Exemplu:

```
DEFINE FILE 5(300,10,E,J)
J = 10
READ (5'J,100)
100 FØRMAT (2F5.2)
```

Pentru înregistrarea pe disc se foloseşte instrucţiunea

WRITE (*n ' e, m*) listă

Semnificaţia literelor este aceeaşi cu cea de la instrucţiunea de citire.

Exemplu:

```
DEFINE FILE 5(300,10, U,K)
DIMENSIØN Z(100)
. . . . . .
WRITE (5'1) (Z(I), I = 1,100)
```

Se va transfera pe discul 5 conţinutul celor 100 de celule care compun vectorul Z. Deoarece informaţia se transmite fără format, înregistrările sînt compuse din cîte 10 cuvinte. Se vor forma deci 10 înregistrări.

A patra instrucţiune utilizată la discuri este

FIND(*n'e*)

şi are efectul de a căuta pe discul *n* începutul înregistrării *e*.

Ea este utilizată împreună cu instrucţiunile de citire pentru a aduce din timp capul de citire în dreptul poziţiei din care urmează să se citească, cîştigîndu-se astfel timp. Între ea şi instrucţiunea de citire se pot intercala orice alte instrucţiuni care vor fi executate simultan cu căutarea înregistrării *e*.

1.3.2. INSTRUCŢIUNI DE ATRIBUIRE

1.3.2.1. Expresii aritmetice

Aceste instrucţiuni corespund relaţiilor matematice efective necesare pentru executarea calculelor dorite. Operatorul aritmetic de atribuire corespunde semnului = din aritmetică, însă semnificaţia lui este ciuntită în sensul că are ca efect numai substituirea valorii variabilei din stînga semnului cu valoarea variabilei sau expresiei aritmetice din dreapta semnului.

Exemplu:

$a = b$

$a = b \cdot c + d\left(\frac{3 + f}{4}\right)^2$

În limbajul FØRTRAN pentru operaţiile aritmetice se folosesc următoarele semne:

\+ — adunare
— — scădere
* — înmulţire
/ — împărţire
** — ridicare la putere

La executarea unei instrucţiuni de atribuire, întîi se calculează valoarea expresiei din dreapta operatorului de atribuire şi apoi valoarea se transferă variabilei de la stînga lui. La executarea calculelor din dreapta operatorului = se ţine cont de următoarele reguli:

— întîi se execută parantezele cele mai interioare;
— ordinea de executare a operaţiilor aritmetice este:

1. **
2. *,/
3. +, —

— explorarea expresiilor aritmetice se face de la stînga la dreapta;
— cînd se întîlnesc consecutiv două operaţii cu acelaşi ordin de prioritate, se execută întîi cea mai din stînga şi de abia după aceea continuă explorarea expresiei.

Să considerăm a doua expresie din exemplul de mai sus. Transcrierea sa în limbaj FØRTRAN va fi:

```
A = B*C + D* ((3.0 + F) /4.0) **2
```

Conform regulilor de mai sus, executarea calculelor va avea loc în ordinea următoare: întîi B * C, apoi paranteza (3,0 + + F), se va împărţi rezultatul acesta la 4, noul rezultat se va ridica la puterea 2, va fi înmulţit cu D şi apoi se va aduna cu rezultatul primei operaţii. Rezultatul final va fi stocat în memorie în spaţiul rezervat variabilei A.

De reţinut că în FØRTRAN nu are sens ca în stînga semnului = să se afle tot o expresie.

La scrierea expresiilor aritmetice la unele calculatoare, trebuie respectate reguli de omogenitate a expresiei, reguli care cer ca toate variabilele şi constantele din expresie să fie de ace-

Scanned with OKEN Scanner

laşi tip. Dacă este necesar să se utilizeze într-o expresie mărimi mixte, trebuie folosite funcţiile utilitare de conversie dintr-un tip în altul:

FLØAT(...) — conversie întreg-real
IFIX(....) — conversie real-întreg
DBLE(....) — conversie real-dublă precizie
DFLØAT(..) — conversie întreg-dublă precizie
SNGL(....) — conversie dublă precizie-real

Cu ajutorul acestor funcţii putem forma expresii de forma:

```
DØUBLE PRECISIØN R,G
. . . . . .
F = P + FLØAT(L) /SNGL(R + G)
```

unde R, G au fost declarate de dublă precizie.

La calculatoarele moderne, de exemplu la IBM 360, în cadrul limbajului FØRTRAN nivel H este permis orice tip de mixaj, rezultatul avînd tipul variabilei prioritare, seria de prioritate fiind Complex, Dublu precizie, real, întreg.

Pentru operaţia de ridicare la putere nu sînt valabile regulile de omogenitate expuse mai sus; este permisă ridicarea unui real la o putere întreagă.

În cazul FØRTRAN nivel H se admit pentru exponenţiere următoarele combinaţii:

— întreg, real, dublă precizie la putere, întreg sau real;
— complex la putere întreg.

Semnul = are şi el efect în ceea ce priveşte conversia tipurilor. Dacă o relaţie de atribuire are forma

```
A = E
```

unde E este o expresie oarecare, iar A a fost declarat de un tip oarecare, indiferent de tipul în care rezultă valoarea lui E, acesta va fi convertit în tipul lui A şi stocat în memorie ca atare.

Exemple:

A = J + 5 calculat în I, convertit în R.

I = B ** 2 — C calculat în R, convertit în I.

Similar au loc conversii din complex în întreg, dublă precizie în real, sau dublă precizie în întreg.

La alcătuirea expresiilor aritmetice se pot folosi o serie de funcţii utilitare care corespund unor funcţii de tipul sinus, cosinus, arcsinus, sinus hiperbolic, logaritmi, exponenţiale etc.

O listă completă a acestor funcţii este dată în tabelul 1.1.

Scanned with OKEN Scanner

TABEL 1.1 **FUNCȚII STANDARD**

SIMBOL	Semnificație	Nr. de argumente	Tipul argumentului	Tipul funcției
ABS	Valoarea absolută a unui număr	1	Real	Real
IABS		1	Întreg	Întreg
DABS		1	Dublă precizie	Dublă precizie
AINT	Semnul lui a înmulțit cu cel mai mare întreg $\leqslant \lvert a \rvert$	1	Real	Real
INT		1	Real	Întreg
IDINT		1	Dublă precizie	Întreg
NINT	Semnul lui a înmulțit cu cel mai apropiat întreg	1	Real	Întreg
ANINT		1	Real	Real
AMØD	Restul împărțirii a două numere a_1, a_2.	2	Real	Real
MØD		2	Întreg	Întreg
AMAXO	Alegerea celui mai mare din n numere. Max(a_1, a_2, ...)	$\geqslant 2$	Întreg	Real
AMAX1		$\geqslant 2$	Real	Real
MAXO		$\geqslant 2$	Întreg	Întreg
MAX1		$\geqslant 2$	Real	Întreg
DMAX1		$\geqslant 2$	Dublă precizie	Dublă precizie
AMINO	Alegerea minimului a n numere. Min(a_1, a_2, ...)	$\geqslant 2$	Întreg	Real
AMIN1		$\geqslant 2$	Real	Real
MINO		$\geqslant 2$	Întreg	Întreg
MIN1		$\geqslant 2$	Real	Întreg
DMIN1		$\geqslant 2$	Dublă precizie	Dublă precizie
FLØAT	Conversie din întreg în real	1	Întreg	Real
IFIX	Conversie din real în întreg	1	Real	Întreg
SIGN	Transferă semnul lui a_2 la a_1	2	Real	Real
ISIGN		2	Întreg	Întreg
DSIGN		2	Dublă precizie	Dublă precizie
DIM	Diferență pozitivă	2	Real	Real
IDIM		2	Întreg	Întreg
SNGL	Conversie dublă precizie-real	1	Dublă precizie	Real
REAL	Partea reală a unui argument complex	1	Complex	Real
AIMAG	Partea imaginară a unui argument complex	1	Complex	Real
CMPLX	CMPLX $(a,b) = a + ib$	2	Real	Complex

Scanned with OKEN Scanner

TABEL 1.1 (CONTINUARE)

DBLE	Conversie real în dublă precizie	1	Real	Dublă precizie
CØNJG	Conjugatul unui argument complex	1	Complex	Complex
EXP	Exponențială cu argument real	1	Real	Real
DEXP	Exponențială cu argument în dublă precizie	1	Dublă precizie	Dublă precizie
CEXP	Exponențială cu argument complex	1	Complex	Complex
EXP10	Exponențială în baza 10	1	Real	Real
ALØG DLØG CLØG	Logaritm natural	1 1 1	Real Dublă precizie Complex	Real Dublă precizie Complex
ALØG10 DLØG10	Logaritmi în baza 10	1 1	Real Dublă precizie	Real Dublă precizie
SIN DSIN CSIN	Sinus trigonometric al argumentului exprimat în radiani	1 1 1	Real Dublă precizie Complex	Real Dublă precizie Complex
CØS DCØS CCØS	Cosinus trigonometric al argumentului exprimat în radiani	1 1 1	Real Dublă precizie Complex	Real Dublă precizie Complex
TAN	Tangentă trigonometrică	1	Real	Real
CØT	Cotangentă trigonometrică	1	Real	Real
SQRT DSQRT CSQRT	Rădăcină pătrată	1 1 1	Real Dublă precizie Complex	Real Dublă precizie Complex
ASIN ACØS ATAN DATAN	Arcsinus Arccosinus Arctangentă	1 1 1 1	Real Dublă precizie	Real Dublă precizie
ATAN2 DATAN2	Arctangenta unui raport (a_1, A_2)	2 2	Real Dublă precizie	Real Dublă precizie
ACØT	Arccotangentă	1	Real	Real

Scanned with OKEN Scanner

1.3.2.2. Expresii logice

Expresiile logice sînt utilizate în special pentru a crea posibilitate de rupere a secvenţei, adică de a efectua salturi condiţionate peste anumite instrucţiuni din program ca efect al îndeplinirii sau nu a unor condiţii logice impuse anterior.

Pentru formarea expresiilor logice se utilizează două tipuri de operatori: operatori de relaţie şi operatori logici propriu-zişi.

Operatorii din prima categorie sînt cuprinşi în tabelul 1.2.

TABELUL 1.2 **OPERATORI DE RELAŢIE**

Operator de relaţie		Variabila logică A definită prin	Valoarea variabilei A	
.EQ.	= egal	A = B.EQ.C	adevărat pt. fals dacă	B = C B ≠ C
.NE.	≠ neegal	A = B.NE.C	adevărat pt. fals pt.	B ≠ C B = C
.LT.	mai mic	A = B.LT.C	adevărat pt. fals pt.	B < C B ⩾ C
.LE.	mai mic sau egal	A = B.LE.C	adevărat pt. fals pt.	B ⩽ C B > C
.GT.	mai mare	A = B.GT.C	adevărat pt. fals pt.	B > C B ⩽ C
.GE.	mai mare sau egal	A = B.GE.C	adevărat pt. fals pt.	B ⩾ C B < C

Aceşti operatori de relaţie pot acţiona numai asupra unor mărimi de acelaşi tip care însă nu pot fi complexe sau logice.

Operatorii logici propriu-zişi sînt cei cuprinşi în tabelul 1.3. Ei acţionează numai asupra variabilelor logice.

TABELUL 1.3 **OPERATORI LOGICI**

Operator	Semnificaţie	Variabila logică A definită prin	Valoarea variabilă A
.NØT.	negative	A = .NØT.B	fals dacă B adevărat adevărat dacă B fals
.AND.	intersecţie (conjuncţie)	A = B.AND.C	este adevărat numai dacă şi B şi C sînt adevărate
.ØR.	reuniune (disjuncţie)	A = B.ØR.C	este adevărat dacă B sau (şi) C sînt adevărate. Este fals numai dacă şi B şi C sînt false

Scanned with OKEN Scanner

Cu ajutorul operatorilor descrişi mai sus se pot forma diverse expresii logice. Operatorii au următoarele priorităţi de execuţie :

1. Operatori de relaţie
2. .NØT.
3. .AND.
4. .ØR.

Astfel expresiile

```
A.AND.B.ØR..NØT.A.AND..NØT.B
SQRT (R**2-4.0*F*C).GT.O..ØR.K.LE.24.AND..NØT.L2
```

unde A, B, L2 au fost declarate variabile logice, vor fi executate ca :

```
((A.AND.B).ØR. ((.NØT.A). AND. (.NØT.B)))
(((SQRT ((B**2) - ((4.0*F)*C))).GT.O.).ØR. ((K.LE.24). AND. (.NØT.
L2)))
```

Una din erorile ce apar frecvent este confuzia dintre

```
.NØT.A.AND.B
```

şi

```
.NØT. (A.AND.B)
```

Utilitatea expresiilor logice va reieşi mai clar după citirea capitolului privitor la ruperile de secvenţă condiţionate.

1.3.3. RUPERILE DE SECVENŢĂ – INSTRUCŢIUNI DE SALT

Instrucţiunile de salt permit întreruperea suitei naturale de instrucţiuni din program şi permit ramificarea spre o secvenţă situată înaintea sau după instrucţiunea de rupere a secvenţei. Aceste instrucţiuni se pot utiliza şi pentru repetarea sau evitarea unei părţi din program.

Să considerăm ca exemplu un program care calculează suma termenilor pozitivi şi suma termenilor negativi ai unei serii de 100 de numere întregi care se citesc unul cîte unul.

Notăm cu SPLUS şi SMINUS cele două sume şi cu TERM variabila sub al cărei nume vom citi rînd pe rînd cele 100 de numere. Cu I vom nota un contor care ne va servi pentru numărarea termenilor citiţi.

Vom da la început sumelor valoarea zero şi apoi, pe măsură ce vom citi termenii şi ne vom decide asupra semnului lor, îi

Scanned with OKEN Scanner

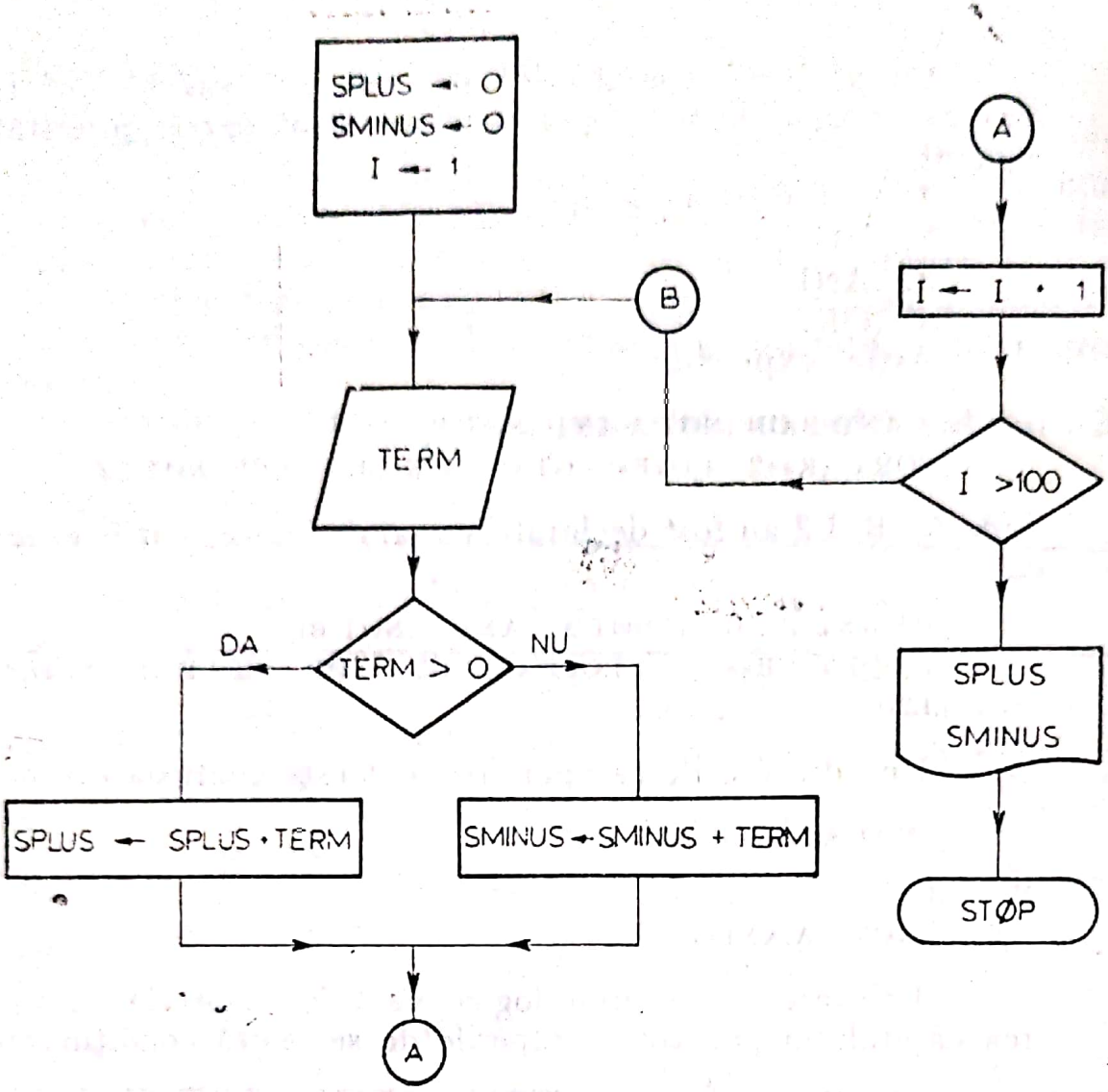


Fig. 1.2.

vom aduna la una sau cealaltă sumă. Cînd valoarea contorului I va atinge valoarea 101 nu vom mai citi şi vom afişa la imprimantă valoarea celor două sume.

Organigrama programului este cea din fig. 1.2

Pentru transcrierea în limbaj FØRTRAN vom folosi două tipuri de instrucţiuni de salt:

instrucţiuni de salt necondiţionat;

instrucţiuni de salt condiţionat.

1.3.3.1. Instrucţiuni de salt necondiţionat

Există trei astfel de instrucţiuni:

— GØ TØ (mergi la)

— GØ TØ obligat

— GØ TØ calculat

Prima dintre aceste instrucţiuni se utilizează în forma:

GØ TØ n

unde n este o etichetă reală din program.

De exemplu:

GØ TØ 35

determină continuarea programului din dreptul instrucţiunii avînd eticheta 35.

Cea de a doua instrucţiune are forma:

GØ TØ J, $(n_1, n_2, \ldots, n_k)$

unde J este o variabilă întreagă oarecare, iar n_i — etichete reale din program.

Această instrucţiune trebuie să fie precedată în program de o instrucţiune de forma:

ASSIGN m TØ J

unde m trebuie să fie identic cu una din etichetele n_i.

Această ultimă instrucţiune (care are semnificaţia „dă lui J valoarea m") se utilizează numai în conexiune cu GØ TØ obligat. Faptul că instrucţiunea ASSIGN ... TØ precede instrucţiunea de salt înseamnă că ea trebuie să fie EXECUTATĂ înaintea acesteia şi nu că ar trebui să se afle înaintea ei în program.

A treia variantă de instrucţiune de salt necondiţionat are forma:

GØ TØ $(n_1, n_2 \ldots, n_k)$, J

unde n_i sînt etichete reale din program, iar întregul J se calculează de obicei cu o relaţie simplă. Valoarea lui J trebuie să fie cuprinsă între 1 şi k. Dacă J va avea valoarea i, atunci se va produce saltul la instrucţiunea cu eticheta n_i.

De exemplu:

GØ TØ (11,46,12,37), KT

— pt. KT = 1 salt la eticheta 11
KT = 3 salt la eticheta 12

sau:

GØ TØ (17,27,27,31), LS

pt. LS = 2 şi LS = 3 se va produce saltul la aceeaşi instrucţiune, care are eticheta 27.

Scanned with OKEN Scanner

1.3.3.2. Instrucţiuni de salt condiţionat

Există două astfel de instrucţiuni :

IF (dacă) aritmetic

IF logic

Prima dintre aceste instrucţiuni este de forma :

IF (expresie aritmetică sau variabilă) n_1, n_2, n_3

Ea are următoarele proprietăţi :

— dacă expresia sau variabila din paranteză este negativă, se continuă programul de instrucţiune cu eticheta n_1 ;

— dacă expresia sau variabila este de valoare zero, se continuă de la instrucţiunea cu eticheta n_2 ;

— dacă valoarea expresiei sau variabilei este pozitivă, execuţia programului se continuă de la instrucţiunea cu eticheta n_3.

A doua instrucţiune de salt condiţionat este de forma :

IF (expresie sau variabilă logică) instrucţiune executabilă.

Instrucţiunea înscrisă după paranteză poate fi de orice fel, exceptînd :

— o instrucţiune IF logică ;

— o instrucţiune de ciclare.

Această instrucţiune are următoarele proprietăţi :

— dacă expresia din paranteză este adevărată, se execută întîi instrucţiunea executabilă de după paranteză ;

— dacă expresia din paranteză este falsă, se execută instrucţiunea care urmează în program, după instrucţiunea de salt.

Exemple :

```
IF (A.LE.B) C = SQRT((B – A)/247.3)
IF((F.LT.3.5).ØR.Z.GT.X) GØ TØ 39
IF(X.NE.A) IF(A)17,31,27
```

Acest ultim exemplu reprezintă de fapt un caz de ramificare în 4 direcţii ; trei pentru cazurile cînd X ≠ A, iar a patra pentru X = A este chiar continuarea secvenţei normale a programului.

Una din regulile esenţiale care trebuie respectată este că instrucţiunea care urmează unei IF aritmetic sau unei instrucţiuni de salt necondiţionat trebuie să aibă neapărat etichetă ; altfel nu se mai poate ajunge la secvenţa ce începe cu acea instrucţiune.

Această eroare este în general semnalată însă în timpul compilării programului.

Scanned with OKEN Scanner

1.3.3.3. Instrucțiuni de control

Instrucțiunile ce urmează servesc la semnalarea sfîrșitului părții executabile a programului și sfîrșitului general al programului.

Instrucțiunea :

STØP

(la unele mașini :

CALL EXIT)

este ultima instrucțiune care trebuie să fie executată.

Ea semnalează sfîrșitul execuției efective (adică terminarea tuturor calculelor și operațiilor anexe de intrare și ieșire) și aduce calculatorul în starea în care poate primi un nou program. Această instrucțiune se poate afla oriunde în program.

Instrucțiunea

END

trebuie să fie fizic ultima instrucțiune din program, rolul e fiind de a semnala sfîrșitul compilării, adică al traducerii automate a programului din limbaj FØRTRAN (sau alt limbaj simbolic) în cod. mașină.

O a treia instrucțiune de control este :

PAUSE *n*

unde *n* este o cifră nu mai mare de 99999.

Această instrucțiune provoacă oprirea calculatorului după imprimarea textului PAUSE *n* la imprimantă. Calculul se reia prin comandă manuală de repornire. Utilizată rar (căci provoacă pierdere de timp), această instrucțiune dă timp operatorului să facă unele operații, ca schimbarea unei benzi magnetice, necesare calculului efectiv.

1.3.3.4. Exemplu

Reluăm exemplul de program a cărui organigramă am construit-o la începutul acestui capitol.

În acea organigramă am utilizat deja simbolul ce reprezintă instrucțiunile de salt condiționat, așa că acum nu vom face decît să atragem atenția că instrucțiunile de salt necondiționat se reprezintă pur și simplu prin săgeți.

Inainte de a da un exemplu de program, vom face unele precizări privind modul de înscriere a instrucțiunilor pe formu-

Scanned with OKEN Scanner

FORTRAN

Program SUMA 100		Indicații de perforare											Pagina 1 din SUMA 100
		Caracter.	(	)	=	+							
Programator GEORGESCU E	Data 15 07 72	Perforare											

C—Ptr. coment. 1 Inst Nr 5	6	INSTRUCTIUNI FORTRAN 7 10 15 20 25 30 35 40 45 50 55 60 65 70 72	Identificare Secvență 73 75 80
		INTEGER SPLUS,SMINUS,TERM	S 1
100		FORMAT(I5)	S 2
200		FORMAT(10X,'SPLUS=',I10/10X,'SMINUS=',I10)	S 3
C		INITIALIZAM SUMELE SI CONTORUL I	S 4
		SPLUS=0	S 5
		SMINUS=0	S 6
		I=1	S 7
C		SE CITESC DATELE SI SE VERIFICA SEMNUL CU UN I.F-ARITMETIC	S 8
1		READ(1,IN 100)TERM	S 9
		IF(TERM)2,,2,3	S 10
2		SMINUS=SMINUS+TERM	S 11
		GØ TØ 4	S 12
3		SPLUS SPLUS+TERM	S 13
4		I=I+1	S 14
C	SE	VERIFICA VALOAREA CONTORULUI I ,CU UN I.F LOGIC	S 15
		IF(I.LE.100)GØ TØ 1	S 16
C	SE	AFISEAZA REZULTATELE	S 17
		WRITE(1Ø,200)SPLUS,SMINUS	S 18
		STØP	S 19
		END	S 20

Fig. 1.3.

Scanned with OKEN Scanner

larele de limbaj FØRTRAN, mod care se păstrează şi la perforarea lor pe cartele.

Fig. 1.3 prezintă un astfel de formular.

Primele 5 coloane servesc pentru înscrierea etichetelor. Numerele se înscriu plasat la dreapta intervalului.

Coloana a 6-a se utilizează pentru a semnala faptul că pe rîndul respectiv se continuă instrucţiunea de pe rîndul anterior. Pentru aceasta se poate folosi orice semn special, literă sau cifră utilizată în limbajul FØRTRAN. Numărul de cartele pe care se poate întinde o instrucţiune variază de la un tip de maşină la altul.

Dacă în coloana 1 se înscrie un C, cartela respectivă este ignorată din punctul de vedere al programului, putîndu-se înscrie pe ea orice comentariu privind etape din program.

Din coloana 7 pînă în 72 se înscriu efectiv instrucţiunile.

În coloanele 73—80 se înscriu de obicei cifre şi semne care să permită identificarea cartelei ca poziţie în program, pentru cazul nefericit în care, scăpînd programul pe jos, s-ar fi amestecat cartelele.

Un exemplu de program corespunzător organigramei de la pagina 35 este înscris chiar în formularul din fig. 1.1.

În acest program se observă că în cadrul instrucţiunilor de I/E nu sînt specificate numerele simbolice ale unităţilor corespunzătoare de I/E. Aceasta permite ca numerele să fie specificate prin două cartele adăugate înaintea oricărei instrucţiuni de I/E. Astfel, la trecerea de pe o maşină de calcul pe o alta care are alte numere simbolice pentru unităţile I/E, nu trebuie modificate toate instrucţiunile I/E, ci doar cele două cartele descrise mai sus.

1.3.4. INSTRUCŢIUNI DE CICLARE

În programul prezentat mai înainte se observă că o anumită parte a programului se execută de 100 de ori. Această ciclare este realizată acolo prin introducerea controlului I şi verificarea continuă a valorii acestuia.

În limbajul FØRTRAN există o instrucţiune care permite realizarea automată a acestei operaţii. Forma ei este:

DØ n J = K_1, K_2, K_3

Ea are următoarea semnificaţie: execută tot ceea ce este cuprins între această instrucţiune şi inclusiv instrucţiunea cu eticheta n pentru fiecare valoare a lui J începînd cu K_1, termi-

Scanned with OKEN Scanner

nînd cu K_2 şi avînd pasul K_3. K_1, K_2 şi K_3 sînt numere întregi. Dacă $K_3 = 1$ nu mai trebuie specificat.

În aceste condiţii, programul anterior (partea de calcul efectiv) ar putea deveni :

```
      DØ 3 J = 1,100
      READ (IN, 100 ) TERM
      IF (TERM)1,1,2
    1 SMINUS = SMINUS + TERM
      GØ TØ 3
    2 SPLUS = SPLUS + TERM
    3 CØNTINUE
```

În această variantă se observă utilizarea unei instrucţiuni neprezentate pînă acum :

```
CØNTINUE
```

Ea este executabilă dar ineficientă, avînd ca singură semnificaţie continuă execuţia. Se utilizează în special în această postură de încheiere a unei bucle de program.

Instrucţiunea cu eticheta *n* nu poate fi :

— o instrucţiune de salt necondiţionat ;
— un IF aritmetic sau logic ;
— o altă instrucţiune de ciclare.

În interiorul unui ciclu pot să existe alte cicluri. Aranjarea acestora este reprezentată în fig. 1.4.

Dăm în continuare cîteva reguli de utilizare a instrucţiunilor de ciclare :

— două cicluri pot avea aceeaşi instrucţiune finală ;
— dintr-un ciclu se poate ieşi oricînd cu ajutorul unei instrucţiuni de salt ;

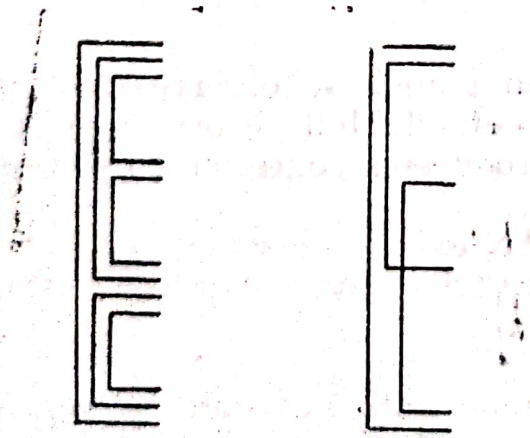


Fig. 1.4.

Scanned with OKEN Scanner

— într-un ciclu nu se poate intra decît pe la instrucţiunea DØ;

— valorile lui K_1, K_2 şi K_3 nu se pot modifica în interiorul ciclului.

În capitolele de aplicaţii se vor întîlni numeroase aplicaţii ale instrucţiunilor de ciclare.

1.4. SUBPROGRAME

După cum s-a văzut înainte, în cadrul limbajului FØRTRAN există o serie de funcţii matematice şi utilitare, care substituie descrierea, de fiecare dată cînd este nevoie, a unor proceduri pentru obţinerea efectelor acestor funcţii. Aceste funcţii sau subprograme se află stocate în memoria calculatorului şi sînt incluse în program ori de cîte ori sînt necesare.

Acest mod de lucru se poate extinde, utilizatorul putînd să scrie subprograme pe care apoi să le poată folosi de mai multe ori în cadrul unui program principal. De asemenea, un subprogram poate fi utilizat în programe diferite, simplificîndu-se astfel munca de programare.

Subprogramele utilizate în FØRTRAN se împart în două categorii:

a) Funcţii

b) Subrutine

1.4.1. FUNCŢII

Funcţiile sînt subprograme care, pe baza unuia sau mai multor argumente transmise de programul principal, calculează o singură valoare care este trimisă înapoi programului principal.

Există două tipuri de funcţii:

— funcţii formulă;

— funcţii subprogram propriu-zise (denumite simplu funcţii).

1.4.1.1. Funcţii formulă

Acestea sînt expresii ce figurează în programul principal înaintea oricărei instrucţiuni executabile.

Exemplu:

```
R(X) = X**2/(1.-X**3)
F(A,B,C) = (A*C/B**3)*ALØG(C)
```


Scanned with OKEN Scanner

Utilizarea unei funcţii formulă se face prin instrucţiuni de atribuire, de exemplu pentru funcţia F scrisă înainte :

```
Z = F(X1,X2,Y)
```

Lista cuprinsă în paranteza ce urmează numelui funcţiei formulă se numeşte listă de chemare, iar lista care apare în paranteză în cazul instrucţiunii de utilizare a funcţiei se numeşte listă de intrare.

Scrierea acestor funcţii se supune următoarelor reguli :

1. Numele funcţiei se supune regulilor de scriere a numelor de variabile (maximum 6 caractere, primul caracter literă, fără caractere speciale, iniţiala corespunde tipului funcţiei (întreg sau real)).

2. Funcţiile al căror rezultat este exprimat în dublă precizie, complex sau logic, trebuie să aibă numele declarat într-o instrucţiune DØUBLE PRECISIØN, CØMPLEX sau LØGICAL.

3. O funcţie formulă poate utiliza o altă funcţie formulă în definiţia ei, cu condiţia ca aceasta din urmă să fi fost definită anterior.

4. Variabilele din lista de intrare trebuie să corespundă ca tip cu variabilele din lista de apel.

5. Numărul maxim de variabile din lista de apel este uneori limitat, funcţie de calculator.

6. În componenţa expresiei de definire a funcţiilor formulă pot intra şi alte variabile decît cele din lista de apel, cu condiţia ca numele acestora să fie identic cu numele unor variabile reale din program şi ca aceste variabile să aibă valori înainte de utilizarea funcţiei formulă care le conţine.

7. În lista de apel (şi deci nici în definiţie) nu pot apărea variabile indexate.

Exemple de funcţie :

```
LØGICAL Z
Z(X,Y) = X.LT.Y
```

Exemple de posibilităţi de chemare a celei de a doua funcţii :

```
A = Z(2.,7.)
A = Z(R,V)
A = Z(SIN (R-6.), CØS(V))
```


Scanned with OKEN Scanner

1.4.1.2. Funcţii subprogram propriu-zise

Aceste funcţii trebuie scrise ca programe aparte. Ele conţin, la fel ca orice program, instrucţiuni pentru declaraţii de tip, pentru alocarea memoriei şi o serie de instrucţiuni executabile. Rezultatul efectuării calculelor trebuie să fie o valoare unică care va fi stocată ca o variabilă cu numele identic cu numele funcţiei.

Prima cartelă dintr-o funcţie subprogram are următorul conţinut:

```
FUNCTIØN Nume (listă de apel).
```

Numele funcţiei trebuie să se conformeze următoarelor reguli:

a. să conţină numai 6 caractere alfanumerice;

b. să nu conţină semne speciale;

c. să înceapă cu o literă;

d. să înceapă cu I, J, K, L, M, N dacă rezultatul trebuie calculat cu virgulă fixă (număr întreg);

e. să înceapă cu alte litere decît I, J, K, L, M, N dacă rezultatul trebuie calculat cu virgulă flotantă (număr real);

f. să nu fie identic cu numele vreunei funcţii din bibliotecă (SIN, EXP, ABS ... etc.);

g. regulile d, e pot fi încălcate dacă funcţia apare într-o declaraţie de tip;

Exemplu:

```
REAL FUNCTIØN IKA (...)
```

h. funcţiile cu rezultat logic, complex sau în dublă precizie trebuie să figureze neapărat într-o declaraţie de tip.

Ultima instrucţiune executată dintr-o funcţie subprogram trebuie să fie:

```
RETURN
```

Această instrucţiune determină trimiterea rezultatului calculelor înapoi la programul principal.

Ultima instrucţiune a unei funcţii este:

```
END
```

Din lista de apel a unei funcţii subprogram pot face parte:

a. nume de variabile întregi şi reale;

b. constante întregi şi reale;

c. expresii aritmetice;

d. nume de funcţie FØRTRAN din bibliotecă;

Scanned with OKEN Scanner

e. numele unei alte funcţii;
f. numele unei subrutine;
g. numele unei matrici (dimensionate în prealabil);
h. serii alfanumerice precedate de *n*H (unde *n* este numărul de semne din serie).

În cazurile e, f în programul care cheamă funcţia trebuie să existe o instrucţiune:

EXTERNAL Nume de funcţii şi subrutine

în care să figureze numele funcţiilor şi subrutinelor ce apar ca elemente ale listei de chemare a funcţiei subprogram.

În cazul g, matricea trebuie să fie dimensionată şi în programul care cheamă, şi în funcţia subprogram.

De exemplu: o funcţie care calculează suma termenilor diagonali ai unei matrici pătrate.

Programul principal:

```
DIMENSIØN A(10,10)
. . . . . .
. . . . . .
S = SD(A,N)
. . . . . .
. . . .  . .
```

unde N × N este dimensiunea unei matrici cu care se lucrează în mod real. (N = 10 este valoarea maximă).

Funcţie

```
FUNCTIØN SD(X,K)
DIMENSIØN X(K,K)
SD = 0,0
DØ 1 I = 1,K
1 SD = SD + X(I,I)
RETURN
END
```

Instrucţiunea DIMENSIØN se poate scrie nedefinit numai în funcţii şi subrutine şi nu în toate variantele de FØRTRAN.

1.4.2. SUBRUTINE

O subrutină se deosebeşte de o funcţie prin faptul că poate trimite înapoi programului principal mai multe date.

Prima cartelă va conţine de astă dată:

SUBRØUTINE Nume (listă de apel)

În acest caz, lista de apel conţine şi numele variabilelor sau matricilor care vor fi trimise înapoi programului principal. Lista de apel a unei subrutine, spre deosebire de cea a unei funcţii, poate să fie şi vidă, dacă subprogramul nu are o funcţie ma-

Scanned with OKEN Scanner

tematică sau dacă comunicarea cu programul principal este realizată integral prin bloc comun (CØMMØN). O regulă care se desprinde din aceasta este că variabilele din lista de apel nu au voie să figureze într-o instrucţiune CØMMØN.

Pentru chemarea unei subrutine se utilizează instrucţiunea:

```
CALL numele subrutinei (listă de intrare)
```

În lista de intrare vor figura de astădată nume de variabile, matrici etc., din programul care cheamă.

În partea de aplicaţii a cărţii se vor întîlni multe exemple de utilizare a subrutinelor.

Ultima instrucţiune executată este la fel ca la funcţii: RETURN, iar ultima instrucţiune din subprogram trebuie să fie END.

În cazul subrutinelor este posibilă aşa-numita întoarcere multiplă (la IBM 360, UNIVAC).

În acest caz, prima cartelă din subrutină va fi, de exemplu:

```
SUBRØUTINE GAMA (A,B,*,*)
```

iar instrucţiunea de chemare:

```
CALL GAMA (X,Y,*37,* 317)
```

unde 37 şi 317 sînt două etichete reale din programul care cheamă.

Subrutina va conţine o instrucţiune de forma:

```
RETURN K
```

De exemplu:

```
SUBRØUTINE GAMA(A,B,*,*)
. . . . . .
RETURN 2
. . . . . .
RETURN ITA
. . . . . .
END
```

Întoarcerea prin RETURN 2 va provoca revenirea în programul principal la instrucţiunea cu eticheta 317.

Întoarcerea prin RETURN ITA va avea următorul efect:

— dacă ITA = 2, întoarcere la eticheta 317;

Scanned with OKEN Scanner

— dacă ITA = 1, întoarcere la eticheta 317;
— dacă ITA = 0, întoarcere la instrucţiunea ce urmează după instrucţiunea de chemare a subrutinei.

1.5. OPERATORUL DATA

Acest operator serveşte la includerea în program a unor date, de orice tip, care sînt permanent necesare programului, indiferent de problema particulară care se soluţionează.

Să considerăm de exemplu un vector A (20). Un procedeu de a da valori lui A ar fi:

```
DIMENSIØN A(20)
A(1) = 2.
A(2) = 7.14
. . . . . .
. . . . . .
A(20) = 12.5
```

Acest procedeu este dezavantajos, pentru că provoacă pierdere de timp (pentru operarea atribuirilor) şi pierdere de memorie (20 celule pentru valorile lui A, 20 celule în care se găsesc iniţial constantele şi în plus celule pentru stocarea celor 20 de instrucţiuni propriu-zise).

Există un alt mijloc mai economic de a „crea" astfel de constante:

```
DATA (A(I), I = 1,20)/2.,7.14, .....,12,5/
```

În acest caz, valorile se creează direct în momentul încărcării programului pentru execuţie.

Forma generală a instrucţiunii este:

```
DATA listă 1/listă de valori numerice/, listă 3/ listă de valori numerice/
```

Aceasta nu este un ordin executabil. Constantele introduse astfel pot fi de orice tip, inclusiv alfanumerice.

De exemplu:

```
DATA KATØ/4H****/
```

adică constanta KATØ este formată din 4 asteriscuri.

Este posibil să se repete de mai multe ori aceeaşi constantă folosind semnul de înmulţire.

Scanned with OKEN Scanner

Exemplu:

```
DATA (X(I),I = 1,4)/2*3.14,2*1.72/
```

are ca efect

X(1) = 3.14
X(2) = 3.14
X(3) = 1.72
X(4) = 1.72

La scrierea instrucţiunii trebuie respectată coincidenţa dintre tipul constantelor şi modul de scriere a cifrelor efective.

Exemplu:

```
DATA A,B,I/3,7.,14/
```

conţine o greşeală, căci A este real, iar valoarea sa este scrisă ca întreg (3).

Constantele astfel introduse nu pot fi însă incluse în blocuri comune. Pentru a se putea realiza acest lucru trebuie folosite un fel de subprograme de date:

```
BLØCK DATA
```

Subprogramele au următoarea organizare:
— prima cartelă este: BLØCK DATA
— ultima instrucţiune este: END
— nu conţin instrucţiuni executabile;
— singurele instrucţiuni admise sînt:

```
INTEGER     DØUBLE PRECISIØN
CØMPLEX     REAL    LØGICAL
DIMENSIØN   CØMMØN
```

Exemplu:

```
BLØCK DATA
CØMMØN A,B,I,Z(10)
DATA A,B,I, (Z(I),I = 1,10)/7.,— 3., 19,10*12345.6/
END
```

CAPITOLUL 2.

METODE NUMERICE APLICATE

În foarte multe domenii ale ştiinţei şi tehnologiei, pentru soluţionarea problemelor ce apar, nu este posibilă utilizarea unor metode matematice exacte. Acest lucru are două cauze posibile: metoda exactă este foarte laborioasă sau metoda exactă nu există. De exemplu: rezolvarea unui sistem de ecuaţii liniare, cu peste 10 ecuaţii, folosind metoda Cramer, este deosebit de dificilă, iar rezolvarea unei ecuaţii de ordin mai mare decît 2 este posibilă, prin metode exacte, numai în anumite cazuri particulare. O altă problemă rareori soluţionabilă exact este, de exemplu, integrarea unor ecuaţii diferenţiale care nu se încadrează în tipurile clasice descrise în manuale şi care în practică se întîlnesc începînd de la calculul unui reactor chimic şi pînă la ecuaţiile de reintrare în atmosferă a navelor cosmice.

Pentru efectuarea acestor calcule sînt folosite o serie de metode aproximative, metode care fac obiectul a ceea ce matematicienii numesc analiză numerică. În general, aceste metode se bazează pe aproximări succesive ale soluţiei exacte. Cît de departe se merge cu aproximarea, sau mai bine zis cît de mult merită să ne apropiem de soluţia reală este o problemă care depinde de cazul studiat, în principal de precizia datelor experimentale pe baza cărora s-a iniţiat calculul. Un exemplu semnificativ în acest sens este oferit de calculul energiei stării fundamentale a atomului de heliu, unde valoarea calculată (cu cele mai speciale metode) are cu cîteva zecimale mai mult decît cifra experimentală. În special, în acest domeniu al chimiei teoretice, există o controversă între fizicienii care se ocupă cu fizica moleculară şi chimiştii pasionaţi de acelaşi subiect, primii încercînd elaborarea unor metode care să permită prezicerea prin calcul a unor proprietăţi fizice ale moleculei, foarte exact determinabile, iar ceilalţi mulţumindu-se cu o acurateţe „chimică", adică cu metode ale căror rezultate să fie în acord cu proprietăţi macroscopice ale substanţelor, ca reactivitatea chimică, căldura de formare etc.

Problema gradului de acurateţe al unei metode are şi un aspect economic: cu cît se cere o precizie mai mare, timpul de calcul se lungeşte.

Scanned with OKEN Scanner

În acest capitol sînt expuse principalele metode ale analizei numerice, utilizate în calcule folosite în chimie. Metodele au fost grupate pe baza operației exacte pe care o substituie și sînt însoțite, pe cît a fost posibil, de exemple de programe și de aplicații relativ simple, dar specific chimice.

2.1. INTEGRARE NUMERICĂ

2.1.1. TEORIE GENERALĂ. FORMULE NEWTON-COTES

Integrarea numerică are la bază posibilitatea de a aproxima pe intervale mici funcția de integrat (uneori neintegrabilă analitic) prin funcții mai simple, ușor integrabile. Ca urmare, pot fi integrate numeric și funcții ale căror expresii analitice nici nu sînt cunoscute, dispunîndu-se numai de un grafic sau de un tabel de valori.

După felul în care sînt alese intervalele în care este aproximată funcția de integrat, există două categorii de metode de integrare numerică:

— cu intervale egale — formulele Newton-Cotes;

— cu intervale neegale — formulele de cuadratură Gauss.

În continuare, sînt prezentate numai metode din prima categorie.

Ecuația de bază din care derivă aceste metode este:

$$\int_a^b f(x)\mathrm{d}x = h\int_0^{\bar{\alpha}} \left(f(x_0) + \bar{\alpha}\cdot f(x_0) + \frac{\bar{\alpha}(\alpha-1)}{2!}\Delta^2 f(x_0) + \dots\right.$$

$$\left. \dots + \frac{\bar{\alpha}(\bar{\alpha}-1)\dots(\bar{\alpha}-n+1)}{n!}\Delta^n f(x_0)\right)\mathrm{d}\bar{\alpha}, \qquad (2.1.1)$$

unde:

$a = x_0$, b — limitele intervalului de aproximare

$\bar{\alpha} = (b-a)/h$

h = pasul de integrare

Δ = operator diferență-înainte cu următoarea definiție

$$\Delta f(x_0) = f(x_0+h) - f(x_0)$$

$$\Delta^2 f(x_0) = \Delta(\Delta F(x_0)) = f(x_0+2h) - 2f(x_0+h) + f(x_0).$$

Efectuînd integrarea din partea dreaptă a ecuației (2.1.1) obținem:

$$\int_a^b f(x)\mathrm{d}x = h(\bar{\alpha} f(x_0) + \frac{\bar{\alpha}^2}{2}\Delta f(x_0) + \left(\frac{\bar{\alpha}^3}{6} - \frac{\bar{\alpha}^2}{4}\right)\Delta^2 f(x_0) +$$

$$+ \left(\frac{\bar{\alpha}^4}{24} - \frac{\bar{\alpha}^3}{6} + \frac{\bar{\alpha}^2}{6}\right)\Delta^3 f(x_0) + \ldots\Big) \qquad (2.1.2)$$

Pentru integrarea propriu-zisă se procedează în felul următor:

— se împarte domeniul pe care se efectuează integrarea într-un număr oarecare de intervale;

— se alege o formulă de tipul (2.1.2) (dînd lui $\bar{\alpha}$ o valoare oarecare, 1,2,3...) și se aplică pe fiecare interval;

— se adună rezultatele obținute pentru fiecare interval, rezultînd astfel integrala totală.

Cele mai uzuale formule de tipul (2.1.2) sînt:

— regula trapezului:

$$\bar{\alpha} = 1 \quad \int_{x_0}^{x_1} f(x)\,\mathrm{d}x = \frac{h}{2}(f(x_0) + f(x_1)) \qquad (2.1.3a)$$

— regula lui Simpson (vezi fig. 2.1.):

$$\bar{\alpha} = 2 \quad \int_{x_0}^{x_2} f(x)\mathrm{d}x = \frac{h}{3}(f(x_0) + 4f(x_1) + f(x_2)). \qquad (2.1.3b)$$

— a doua regulă a lui Simpson (vezi fig. 2.2.):

$$\bar{\alpha} = 3 \quad \int_{x_0}^{x_3} f(x)\mathrm{d}x = \frac{3h}{8}(f(x_0) + 3(f(x_1) + f(x_2)) + f(x_3)). \qquad (2.1.3c)$$

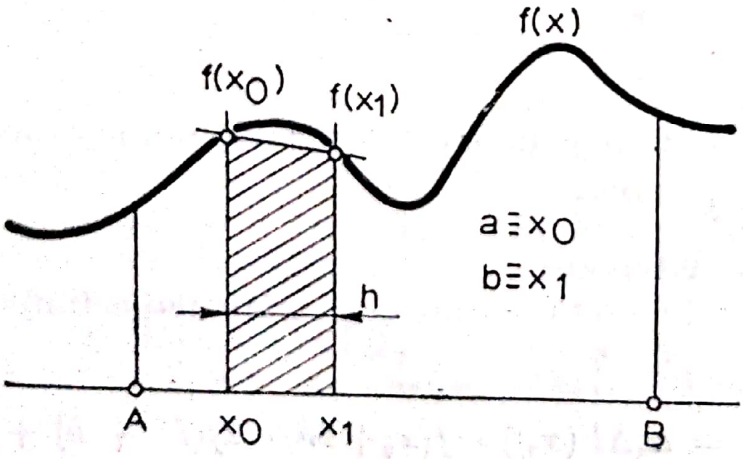


Fig. 2.1.

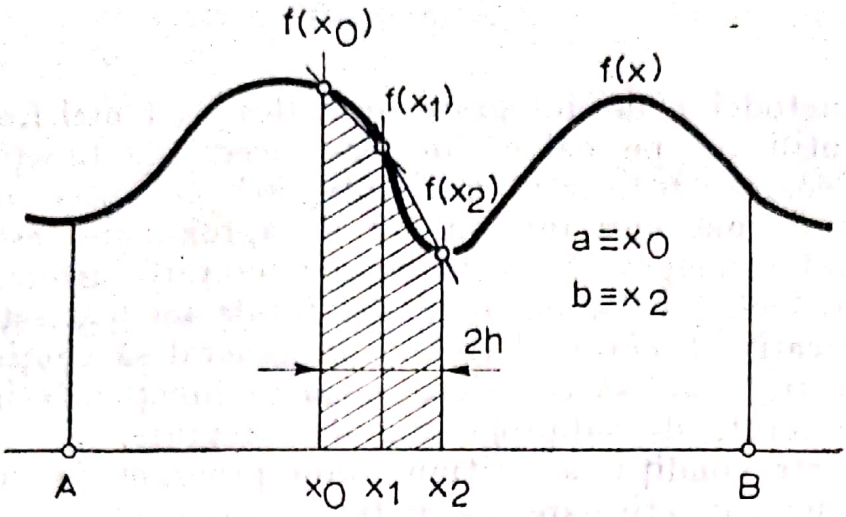


Fig. 2.2.

După cum se observă din ecuaţiile (2.1.3a-c) pentru evaluarea integralelor cu ajutorul lor este necesară cunoaşterea valorilor funcţiei de integrat la capetele intervalelor de aproximare. Din acest motiv, metodele expuse mai sus se numesc formule Newton-Cotes închise.

Există o serie de formule, similare cu acestea, dar care nu necesită pentru utilizare cunoaşterea valorilor funcţiei de integrat la extremităţile intervalelor de aproximare. Aceste metode se numesc Newton-Cotes deschise. Ecuaţia de bază din care provin este:

$$\int_a^b f(x)\mathrm{d}x = h(\bar{\alpha}f(x_1) + \left(\frac{\bar{\alpha}^2}{2} - \bar{\alpha}\right)\Delta f(x_1) + \left(\frac{\bar{\alpha}^3}{6} - \frac{\bar{\alpha}^2}{2} + \bar{\alpha}\right)$$

$$\Delta^2 f(x_1) + \ldots). \qquad (2.1.4)$$

Variantele cele mai utilizate sînt:

$$\bar{\alpha} = 2 \quad \int_{x_0}^{x_2} f(x)\mathrm{d}x = 2hf(x_1) \qquad (2.1.5a)$$

$$\bar{\alpha} = 3 \quad \int_{x_0}^{x_3} f(x)\mathrm{d}x = \frac{3h}{2}(f(x_1) + f(x_2)) \qquad (2.1.5b)$$

$$\bar{\alpha} = 5 \quad \int_{x_0}^{x_5} f(x)\mathrm{d}x = \frac{5h}{24}(11f(x_1) + f(x_2) + f(x_3) + 11f(x_4)) \qquad (2.1.5c)$$

Scanned with OKEN Scanner

În cazul metodelor de integrare numerică, cea mai frecventă formă de utilizare pe calculator este aceea de funcție subprogram. Deoarece rezultatul unei integrări numerice depinde foarte mult de mărimea intervalelor de aproximare, este bine ca programul principal să efectueze o micșorare sistematică a acestora, pînă cînd valoarea integralei totale nu mai este afectată semnificativ. Programul trebuie în general să conțină și o funcție-subrutină care să calculeze valoarea funcției de integrat în punctele cerute de subprogramul de integrare.

În aceste condiții, algoritmul unui program de integrare numerică a unei funcții oarecare ar fi:

1. Citește datele inițiale:
   — limitele de integrare;
   — interval de aproximare inițial;
   — precizie în valoarea integralei totale.
2. Inițializează integrala totală dîndu-i valoarea zero.
3. Transferă valoarea integralei unei mărimi auxiliare pentru a fi conservată pînă la obținerea unei noi valori cu alt interval de aproximare.
4. Calculează pasul de integrare funcție de metoda de integrare utilizată (pas = interval de aproximare $/\alpha$).
5. Efectuează integrarea între limitele prestabilite.
6. Compară valoarea integralei totale cu valoarea transferată mărimii auxiliare; dacă s-a atins precizia cerută, continuă de la etapa nr. 8; dacă nu s-a atins precizia dorită, continuă cu etapa nr. 7.
7. Micșorează intervalul de aproximare; continuă de la etapa nr. 3.
8. Scrie valoarea intervalului de aproximare final și valoarea integralei totale.

Se dau în continuare listele a două funcții-subprogram, FUNCTIØN TRAP (XØ, H) și FUNCTIØN SIMPS2(XØ,H), care corespund metodei trapezului și respectiv celei de a doua reguli a lui Simpson.

```
FUNCTIØN TRAP(XØ,H)
X1 = XØ + H
TRAP = (F(XØ) + F(X1))*H/2.0
RETURN
END
```


Scanned with OKEN Scanner

```
FUNCTIØN SIMPS2(XØ,H)
X1 = XØ + H
X2 = X1 + H
X3 = X2 + H
SIMPS2 = (3.*H/8.)*(F(XØ) + 3.* (F(X1) + F (X2)) + F(X3))
RETURN
END
```

În cazul utilizării acestor subprograme, programul principal trebuie să conţină şi o funcţie-subprogram FUNCTIØN F(X) care să calculeze valoarea funcţiei de integrat.

2.1.3. APLICAŢII. VARIAŢIA CĂLDURII DE REACŢIE CU TEMPERATURA

Valoarea căldurii de reacţie, funcţie de temperatură, este dată de ecuaţia:

$$\Delta H_R = \Delta H_R^0 + \int_{T_0}^{T} \Delta C_p \mathrm{d}T, \tag{2.1.6}$$

unde:

H_R^0 — căldura de reacţie la temperatura T_0;

T — temperatura pentru care se calculează valoarea căldurii de reacţie;

C_p — variaţia căldurii specifice prin reacţie.

Dacă reacţia chimică este

$$a_1A_1 + a_2A_2 + \ldots + a_nA_n \rightarrow b_1B_1 + b_2B_2 + \ldots + b_mB_m,$$

atunci

$$\Delta C_p = \sum_{i=1}^{n} a_i C_{pi} - \sum_{j=1}^{m} b_j C_{pj}. \tag{2.1.7}$$

Căldurile specifice C_p sînt funcţii de temperatură.

Acest tip de calcul este întîlnit ori de cîte ori este necesară proiectarea tehnologică a unui reactor chimic care nu lucrează în regim izoterm.

Schema logică a unui subprogram care efectuează calculul căldurii de reacţie la diverse temperaturi ar fi cea din fig. 2.3.

Scanned with OKEN Scanner

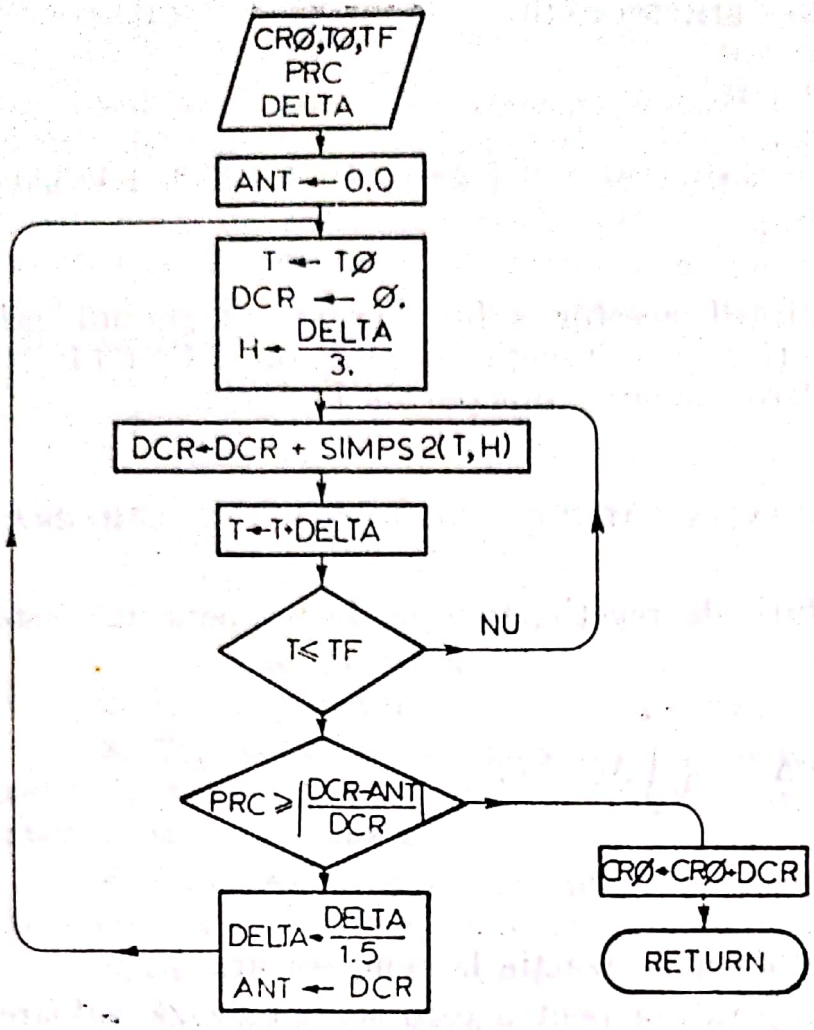


Fig. 2.3.

S-au folosit următoarele notaţii:

DCR — variaţia căldurii de reacţie $= \int_{T_0}^{T} \Delta C_p dT$;

CSR(I) — coeficienţii stoechiometrici ai reactanţilor;
CSP(J) — coeficienţii stoechiometrici ai produşilor;
CPR(I, T) — funcţiile de căldură specifică ale reactanţilor;
CPP(J, T) — funcţiile de căldură specifică ale produşilor;
F(T) — funcţia de căldură specifică — C_p;
NR, NP — numărul de reactanţi şi de produşi;
DELTA — intervalul de aproximare;
H — pasul de integrare;
PRC — precizia;
TØ,TF — limitele intervalului de integrare;
ANT — mărime care serveşte pentru stocarea temporară a valorilor variaţiei căldurii de reacţie;

Scanned with OKEN Scanner

CRØ — căldura de reacţie la temperatura TØ. Tot sub acest nume se obţine şi căldura de reacţie la temperatura TF. Funcţia F(T) va avea următoarea formă :

```
      FUNCTIØN F(T)
      CØMMØN NR,NP,CSR(5),CSP(5)
      F = 0.0
      DØ 1 J = 1, NR
    1 F = F + CSR(J)*CPR(J,T)
      DØ 2 J = 1,NP
    2 F = F — CSP (J)*CPP(J,T)
      RETURN
      END
```

S-a considerat că alegerea valorii 5 pentru numărul maxim de coeficienţi stoechiometrici acoperă majoritatea cazurilor întîlnite în practică. S-a ales blocul comun ca modalitate de transfer spre subrutina F(T) pentru a se menţine secvenţa ei de chemare în forma pe care o necesită subrutina de integrare SIMPS2.

Pentru căldurile specifice ale reactanţilor funcţia CPR va avea forma :

```
      FUNCTIØN CPR(I,T)
      IF(1,2,3,...,NR),I
    1 CPR = funcţia de temperatură a reactantului 1
      GØ TØ 1000
    2 CPR = funcţia de temperatură a reactantului 2
      . . . . . .
      . . . . . .
 1000 RETURN
      END
```

O funcţie similară trebuie să fie scrisă şi pentru produşii de reacţie.

În aceste condiţii, subprogramul care va efectua calculul căldurii de reacţie folosind funcţia SIMPS2 descrisă anterior şi subprogramele-funcţie de mai sus va fi următorul :

```
      SUBRØUTINE VCR (CRØ,TØ, TF, PRC, DELTA)
      ANT = 0.0
    1 T = TØ
      DCR = 0.0
      H = DELTA/3.
    2 DCR = DCR + SIMPS2(T,H)
      T = T + DELTA
      IF(T — TF)2,2,3
```


Scanned with OKEN Scanner

```
3 IF (PRC - ABS ((DCR-ANT)/DCR))4,5,5
4 ANT = DCR
  DELTA = DELTA/1.5
  GO TO 1
5 CRO = CRO + DCR
  RETURN
  END
```

2.2. REZOLVAREA ECUAȚIILOR DIFERENȚIALE ORDINARE

2.2.1. CLASIFICAREA METODELOR

O ecuație diferențială ordinară de ordinul n are forma:

$$F\left(x,\ y,\ \frac{dy}{dx},\ \frac{d^2y}{dx^2},\ \ldots,\ \frac{d^ny}{dx^n}\right) = 0. \tag{2.2.1}$$

Este de ordinul n, căci conține derivata de ordinul n, și se numește ordinară, pentru că în ea apar numai derivate totale (nu apar derivate parțiale).

Pentru a se obține o soluție unică a ecuației (2.2.1) trebuie cunoscute n condiții suplimentare pe care trebuie să le satisfacă soluția. Dacă aceste condiții se referă la mai multe puncte din zona în care ne interesează soluția (preferabil puncte care mărginesc zona), problema se numește problemă cu valori limită. În cazul în care toate condițiile se referă la un singur punct (X_0), spunem ca avem de-a face cu o problemă cu valori inițiale.

Orice ecuație diferențială ordinară de ordinul n se poate transforma într-un sistem de n ecuații de ordinul întîi prin definirea a $n-1$ variabile noi. De exemplu, ecuația Bessel:

$$x^2\frac{d^2y}{dx^2} + x\frac{dy}{dx} + (x^2 - p^2)\,y = 0 \tag{2.2.2}$$

se poate simplifica, făcînd transformarea $z = dy/dx$. Se obține sistemul

$$\begin{cases} \dfrac{dy}{dx} - z = 0 \\ x^2\dfrac{dz}{dx} + xz + (x^2 - p^2)y = 0. \end{cases} \tag{2.2.3}$$

Scanned with OKEN Scanner

În continuare ne vom referi numai la metode de rezolvare a ecuațiilor diferențiale ordinare de ordinul întîi și a unor sisteme compuse din astfel de ecuații.

Ecuația de ordinul întîi are forma:

$$\frac{dy}{dx} = f(x, y). \tag{2.2.4}$$

Soluția căutată $Y(x)$ va trebui să satisfacă și o condiție inițială. O soluție analitică nu se poate obține întotdeauna.

Pentru a se obține o soluție numerică, intervalul în care este definită variabila independentă $[a,b]$ este împărțit în segmente sau pași de lungime h. Numărul de pași va fi:

$$n = (b - a)/h. \tag{2.2.5}$$

Ca efect al rezolvării numerice, valoarea soluției adevărate $Y(x)$ este aproximată pentru n valori ale lui x. Soluția este dată sub forma unui tabel de $(n + 1)$ valori ale lui Y, corespunzătoare celor $(n + 1)$ valori ale variabilei x, $x_i = x_0 + ih$, unde $i = 0, 1, 2 \ldots n$.

Metodele de aproximare ale soluției, atunci cînd se cunoaște valoarea inițială, corespunzătoare lui x_0, se bazează în principal pe două căi de a aborda problema:

— dezvoltarea în serie Taylor a soluției $Y(x)$;

— utilizarea unei formule de integrare de tip Newton-Cotes.

Metodele se pot clasifica în două mari categorii:

— metode pas cu pas — în cadrul acestor metode pentru a calcula Y_{i+1} este nevoie numai de Y_i, x_i;

— metode de mai mulți pași (multi-pas); în cazul acestora sînt necesare mai multe valori x_k, Y_k pentru a realiza pasul $i \to i + 1$.

2.2.2. METODE PAS CU PAS

Dezvoltarea în serie Taylor a soluției $Y(x)$ în jurul punctului inițial (x_0) duce la expresia:

$$Y(x_0 + h) = Y(x_0) + hY'(x_0) + \frac{h^2}{2!} Y''(x_0) + \ldots \tag{2.2.6}$$

Scanned with OKEN Scanner

dar

$$Y'(x_0) = \frac{d}{dx}(Y(x_0)) = f(x_0, Y(x_0))$$

$$Y''(x_0) = \frac{d}{dx}(Y'(x_0)) = f'(x_0, Y(x_0))$$

$$\vdots \qquad \vdots \qquad \vdots$$

Deci:

$$Y(x_0+h) = Y(x_0) + hf(x_0, Y(x_0)) + \frac{h^2}{2!}f'(x_0, Y(x_0)) \ldots \qquad (2.2.7)$$

La evaluarea termenilor superiori din ecuația precedentă trebuie ținut cont de faptul că f este o funcție de două variabile:

$$f' = \frac{df}{dx} = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} \cdot \frac{dy}{dx}. \qquad (2.2.8)$$

Extinzînd ecuația (2.2.7) putem scrie:

$$Y_{i+1} = Y_i + hf_i + \frac{h^2 f_i'}{2!} + \ldots + \frac{h^2 f_i^{(n)}}{n!} + \ldots \qquad (2.2.9)$$

Cea mai simplă metodă, care rezultă din această ecuație, este aceea în care se neglijează toți termenii care conțin pe h la puteri mai mari de 1.

Această metodă se numește metoda Euler:

$$\begin{aligned} Y_1 &= Y(x_0) + hf(x_0, Y(x_0)) \\ Y_{i+1} &= Y_i + hf_i \end{aligned} \qquad (2.2.10)$$

Precizia acestei metode depinde foarte mult de valoarea pasului h.

Metoda se poate extinde foarte ușor la un sistem de ecuații diferențiale:

$$\begin{aligned} \frac{dY_1}{dx} &= f_1(x, Y_1, Y_2, \ldots Y_n) \\ \frac{dY_2}{dx} &= f_2(x, Y_1, Y_2, \ldots Y_n) \\ &\ldots\ldots\ldots\ldots \\ \frac{dY_n}{dx} &= f_n(x, Y_1, Y_2, \ldots Y_n) \end{aligned} \qquad (2.2.11)$$

Scanned with OKEN Scanner

Ecuaţiile (2.2.10) în acest caz devin:

$$Y_{j,1} = Y_j(x_0) + hf_j(x_0, Y_1(x_0), Y_2(x_0), \ldots)$$
$$Y_{j,k+1} = Y_{j,k} + hf_j(x_k, Y_{1,k}, Y_{2,k}, \ldots, Y_{n,k}) \qquad (2.2.12)$$

Pentru a se obţine o precizie satisfăcătoare, în cazul acestei metode, pasul de integrare trebuie micşorat foarte mult. Aceasta duce la o creştere însemnată a timpului de calcul. Din această cauză metoda este destul de puţin utilizată. Se enumeră în continuare cîteva metode, din aceeaşi categorie cu metoda Euler, care sînt mai des folosite deoarece sînt mai precise:

— metoda Euler modificată:

$$Y_{i+1} = Y_i + hf\left(x_i + \frac{h}{2}, \overline{Y}_{i+1/2}\right)$$
$$\overline{Y}_{i+1/2} = Y_i + hf_i \qquad (2.2.13)$$

— metoda Heun sau Euler îmbunătăţită:

$$Y_{i+1} = Y_i + \frac{h}{2}(f(x_i, Y_i) + f(x_{i+1}, \overline{Y}_{i+1})) \qquad (2.2.14a)$$

$$\overline{Y}_{i+1} = Y_i + hf(x_i, Y_i) \qquad (2.2.14b)$$

Această metodă face parte din categoria metodelor predictor-corector, deoarece cu ecuaţia a doua se calculează o primă valoare pentru Y_{i+1} care este apoi corectată cu prima ecuaţie. Ecuaţia corector se poate aplica succesiv de mai multe ori pînă cînd variaţia lui Y_{i+1} scade sub o anumită limită impusă.

— metoda Runge-Kutta de ordinul trei:

$$Y_{i+1} = Y_i + \frac{h}{6}(k_1 + k_2 + k_3)$$
$$k_1 = f(x_i, Y_i) \qquad (2.2.15)$$
$$k_2 = f(x_i + h/2, Y_i + hk_1/2)$$
$$k_3 = f(x_i + h, Y_i + 2hk_2 - hk_1)$$

Scanned with OKEN Scanner

— metoda Runge-Kutta de ordinul patru, varianta Gill:

$$Y_{i+1} = Y_i + \frac{h}{6}\left(k_1 + 2\left(1 - \frac{1}{\sqrt{2}}\right)k_2 + 2\left(1 + \frac{1}{\sqrt{2}}\right)k_3 + k_4\right)$$

$$k_1 = f(x_i, Y_i)$$

$$k_2 = f(x_i + h/2, Y_i + hk_1/2) \qquad (2.2.16)$$

$$k_3 = f\left(x_i + h/2,\ Y_i + \left(\frac{1}{\sqrt{2}} - \frac{1}{2}\right)hk_1 + \left(1 - \frac{1}{\sqrt{2}}\right)hk_2\right)$$

$$k_4 = f\left(x_i + h, Y_i - hk_2/\sqrt{2} + \left(1 + \frac{1}{\sqrt{2}}\right)hk_3\right)$$

Această metodă este utilizată cel mai frecvent, dintre toate metodele pas cu pas.

2.2.3. PROGRAME. METODA HEUN. METODA RUNGE-KUTTA

Programele care se descriu în continuare sînt concepute pentru a putea rezolva sisteme de ecuaţii de tipul (2.2.11). Se folosesc următoarele notaţii:

N — numărul de ecuaţii din sistem;

H — pasul de integrare;

X — variabila independentă x la începutul pasului de integrare;

Y(I) — valoarea variabilei Y_i la începutul unui pas de integrare;

Y1(I) — valoarea variabilei Y_i la sfîrşitul unui pas de integrare;

F(I,X,Y) — subprogram funcţie corespunzător lui $f_i(x, Y_1, Y_2, \ldots)$.

În cazul în care dorim să efectuăm integrarea conform metodei Heun algoritmul unui subprogram care calculează un pas ar fi:

1. Primeşte de la programul principal,

N,H,X,(Y(I), I = 1,N)

2. Folosind ecuaţia (2.2.14b) calculează o primă aproximare pentru Y1(I)

Y1 (I) = Y(I) + H * F(I,X,Y)

Scanned with OKEN Scanner

3. Folosind ecuația corector (2.2.14a) calculează o nouă valoare, care se notează Y2(I)

4. Verifică dacă diferența dintre Y2(I) și Y1(I) este sau nu sub o anumită limită. Dacă da, continuă cu etapa 5; dacă nu, continuă cu etapa 6.

5. Trimite valorile Y1(I) programului principal și oprește calculele.

6. Transferă valorile Y2(I) lui Y1(I) și apoi reia de la etapa 3.

Organigrama corespunzătoare acestui algoritm este cea din fig. 2.4.

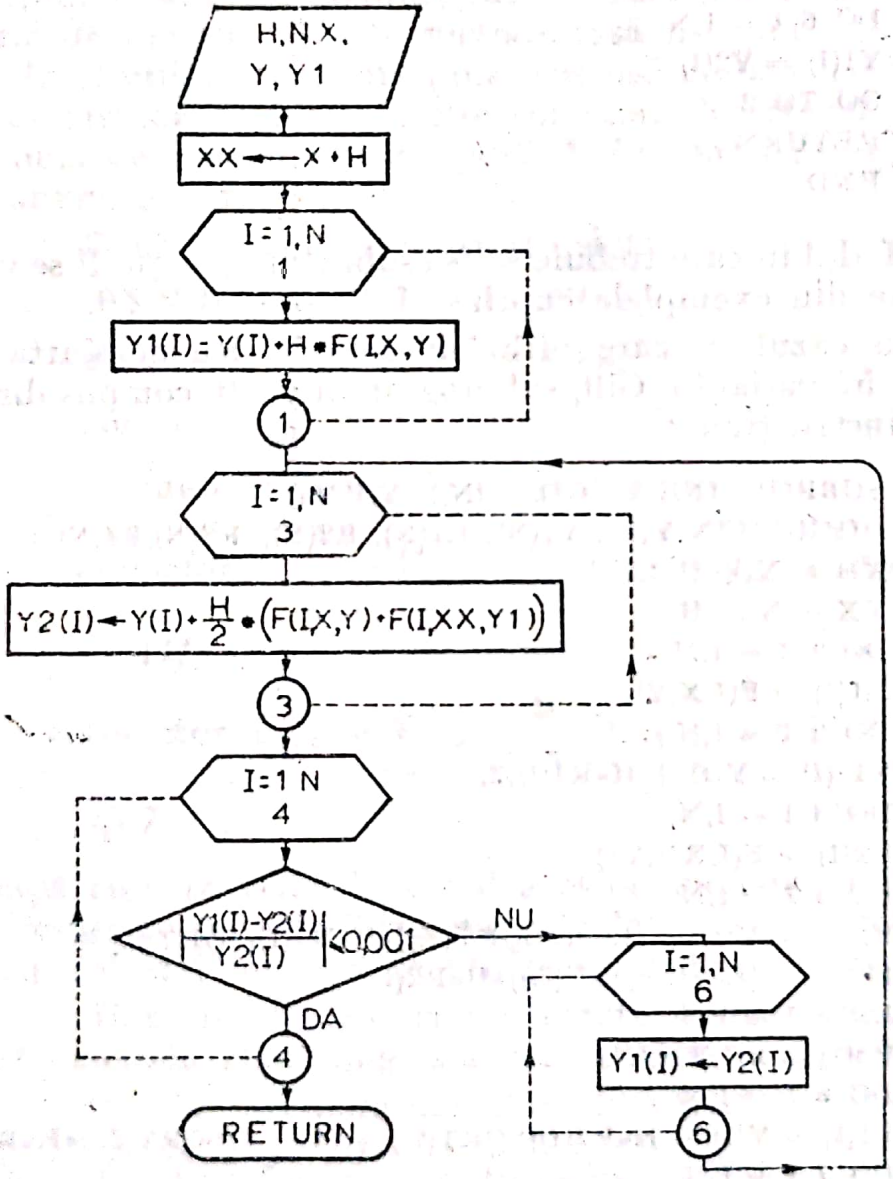


Fig. 2.4.

Subprogramul propriu-zis este compus din următoarele instrucţiuni :

```
      SUBRØUTINE HEUN (N,H,X,Y,Y1)
      DIMENSIØN Y(N), Y1 (N), Y2(N)
      XX = X + H
      DØ 1 I = 1,N
    1 Y1(I) = Y(I) + H*F(I,X,Y)
    2 DØ 3 I = 1,N
    3 Y2(I) = Y(I) + (F(I,X,Y) + F(I,XX,Y1))*H/2.
      DØ 4 I = 1,N
      IF(ABS((Y1(I)-Y2(I))/Y2(I))-0.001)4,4,5
    4 CØNTINUE
      GØ TØ 7
    5 DØ 6 I = 1,N
    6 Y1(I) = Y2(I)
      GØ TØ 2
    7 RETURN
      END
```

Modul în care trebuie scrisă subrutina funcţie F se va vedea mai bine din exemplele cuprinse în capitolul 2.2.6.

În cazul în care utilizăm metoda Runge-Kutta de ordinul 4 în varianta Gill, subprogramul va fi compus din următoarele instrucţiuni :

```
      SUBRØUTINE RKGILL (N,H,X,Y,Y1)
      DIMENSIØN Y(N), Y1(N), R1(N), R2(N), R3(N),R4(N)
      XH = X + H/2.
      XX = X + H
      DØ 1 I = 1,N
    1 R1(I) = F(I,X,Y)
      DØ 2 I = 1,N
    2 Y1 (I) = Y(I) + H*R1(I)/2.
      DØ 3 I = 1,N
    3 R2(I) = F(I,XH,Y1)
      DØ 4 I = 1,N
    4 Y1(I) = Y(I) + (1.-1./SQRT(2.))*H*R1(I)/SQRT(2.) +
     1          (1.-1./SQRT(2.))*H*R2(I)
      DØ 5 I = 1,N
    5 R3(I) = F(I,XH,Y1)
      DØ 6 I = 1,N
    6 Y1(I) = Y(I) - H*R2(I)/SQRT(2.) + (1.+ 1./SQRT(2.))*H*R3(I)
      DØ 7 I = 1,N
```


Scanned with OKEN Scanner

```
7 R4(I) = F(I,XX,Y1)
  DØ 8 I = 1,N
8 Y1(I) = Y(I) + H*(R1(I) + 2.*(1.- 1./SQRT(2.))*(R2(I) + R3(I))
1         +R4(I))/6.
  RETURN
  END
```

2.2.4. METODE CU MAI MULȚI PAȘI

Aceste metode se bazează pe formulele de integrare Newton-Cotes descrise în cap. 2.2.1. Pe baza formulelor deschise s-au obținut o serie de metode, iar pe baza formulelor închise altă serie de metode. Aceste metode însă nu sînt folositeca atare, ele fiind utilizate cuplate (una deschisă cu una închisă) în cadrul unor metode de tip predictor-corector. Corectorul se aplică totdeauna de mai multe ori, pînă cînd variația lui Y_{i+1} scade sub o anumită limită.

Două metode de acest tip sînt:

—metoda Milne de ordinul 4:

$$\text{predictor } Y_{i+1} = Y_{i-3} + 4h(2f_i - f_{i-1} + 2f_{i-2})/3 \quad (2.2.17)$$

$$\text{corector } Y_{i+1} = Y_{i-1} + h(f_{i+1} + 4f_i + f_{i-1})$$

— metoda Milne de ordinul 6:

$$\text{predictor } Y_{i+1} = Y_{i-5} + \frac{3h}{10}(11f_i - 14f_{i-1} + 26f_{i-2} - 14f_{i-3} + 11f_{i-4})$$

$$\text{corector } Y_{i+1} = Y_{i-3} + \frac{2h}{45}(7f_{i+1} + 32f_i + 12f_{i-1} + 32f_{i-2} + 7f_{i-3})$$

După cum se vede, în cazul acestor metode primele 4 iterații și respectiv primele 6 trebuie calculate cu alte metode. În acest scop cel mai frecvent se utilizează metoda Runge-Kutta.

Una dintre cele mai utilizate metode din această clasă este metoda Hamming. Ea are următorul algoritm:

1. Sînt necesare următoarele valori:

$$Y_0, Y_1, Y_2, Y_3, f_0, f_1, f_2, f_3$$

Scanned with OKEN Scanner

Acestea se determină pornind de la Y_0 și f_0, cu ajutorul unei metode pas cu pas. După aceea, etapele 2—6 se aplică succesiv pentru $i = 3,4, \ldots$

2. Soluția precalculată $Y_{i+1,0}$ este dată de predictorul

$$Y_{i+1,0} = Y_{i-3} + \frac{4}{3} h(2f_i - f_{i-1} + 2f_{i-2}) \tag{2.2.18}$$

3. Valoarea aceasta este corectată întîi pe baza presupunerii că în două intervale vecine corecția totală nu diferă prea mult:

$$Y^*_{i+1,0} = Y_{i+1,0} + \frac{112}{121}(Y_{i,k} - Y_{i,0}), \tag{2.2.19}$$

unde

$Y_{i,0}$ — soluția dată de predictor la pasul i;
$Y_{i,k}$ — soluția finală, dată de corector.

Această corecție nu se aplică pentru $i = 3$.

4. Se aplică iterativ corectorul

$$Y_{i+1,j+1} = (1/8)(9Y_i - Y_{i-2} + 3h(f(x_{i+1}, Y_{i+1,j}) + 2f_i - f_{i-1})). \tag{2.2.20}$$

Drept valoare inițială se folosește $Y^*_{i+1,0}$.

Conform teoriei, ar trebui aplicat corectorul pînă cînd s-ar satisface o relație de forma:

$$\left| \frac{Y_{i+1,j+1} - Y_{i+1,j}}{Y_{i+1,j}} \right| < \varepsilon \tag{2.2.21}$$

Practic, dacă lungimea pasului h este bine reglată, este suficientă o singură aplicare a corectorului.

5. Se calculează eroarea de trunchiere a ecuației corectoare

$$e_i = \frac{9}{121}(Y_{i+1,k} - Y_{i+1,0}) \tag{2.2.22}$$

6. Valoarea finală a soluției Y_{i+1} se calculează cu:

$$Y_{i+1} = Y_{i+1,k} - e_i. \tag{2.2.23}$$

Cu această valoare se calculează f_{i+1}.
Calculele se continuă mărind pe i cu 1 și reluînd etapa 2.

Scanned with OKEN Scanner

În cazul în care se aplică metoda Hamming pentru rezolvarea unui sistem de tipul (2.2.11), se folosesc următoarele notaţii:

Y(I,J) — o matrice de dimensiune $4 \times N$ în care sînt stocate iniţial valorile Y_i, Y_{i-1}, Y_{i-2}, Y_{i-3};

FF(K,J) — o matrice de dimensiune $3 \times N$ în care sînt stocate iniţial valorile f_i, f_{i-1}, f_{i-2};

E (J) — un vector în care se stochează cele N erori de trunchiere e_i.

Organigrama subprogramului este cea din fig. 2.5.

S-au făcut următoarele notaţii:

XX(4) — vector în care se stochează valorile variabilei independente X.

F(J,X,Y) — subprogramul funcţiei care calculează valorile f.

YO(J) — vector în care se stochează $Y_{i+1,0}$ pentru a se putea calcula apoi erorile e_i.

Z(J) — vector în care se stochează temporar valorile $Y_{i+1,j+1}$ pentru a putea fi comparate cu $Y_{i+1,j}$ care este stocat în prima linie a matricii Y(I,J).

Programul propriu-zis este compus din următoarele instrucţiuni:

```
      SUBRØUTINE HAMING(N,H,X,E,Y)
      DIMENSIØN Y(4,N), FF(3,N), E(N), Z(N), XX(4), YO(N)
      XX(1) = X + H
      XX(2) = X
      XX(3) = X - H
      XX(4) = X - 2*H
      DØ 2 I = 1,3
      DØ 1 J = 1,N
    1 Z(J) = Y(I,J)
      DØ 2 J = 1,N
      JP1 = J + 1
      FF(I,J) = F(J,XX(JP1),Z)
    2 CØNTINUE
      DØ 3 J = 1,N
      Z(J) = Y(4,J) + 4.*H*(2.*FF(1,J) - FF(2,J) +
     1       +2*FF(3,J))/3.
    3 YØ(J) = Z(J)
      DØ 4 J = 1,N
      Y(4,J) = Y(3,J)
      Y(3,J) = Y(2,J)
```


Scanned with OKEN Scanner

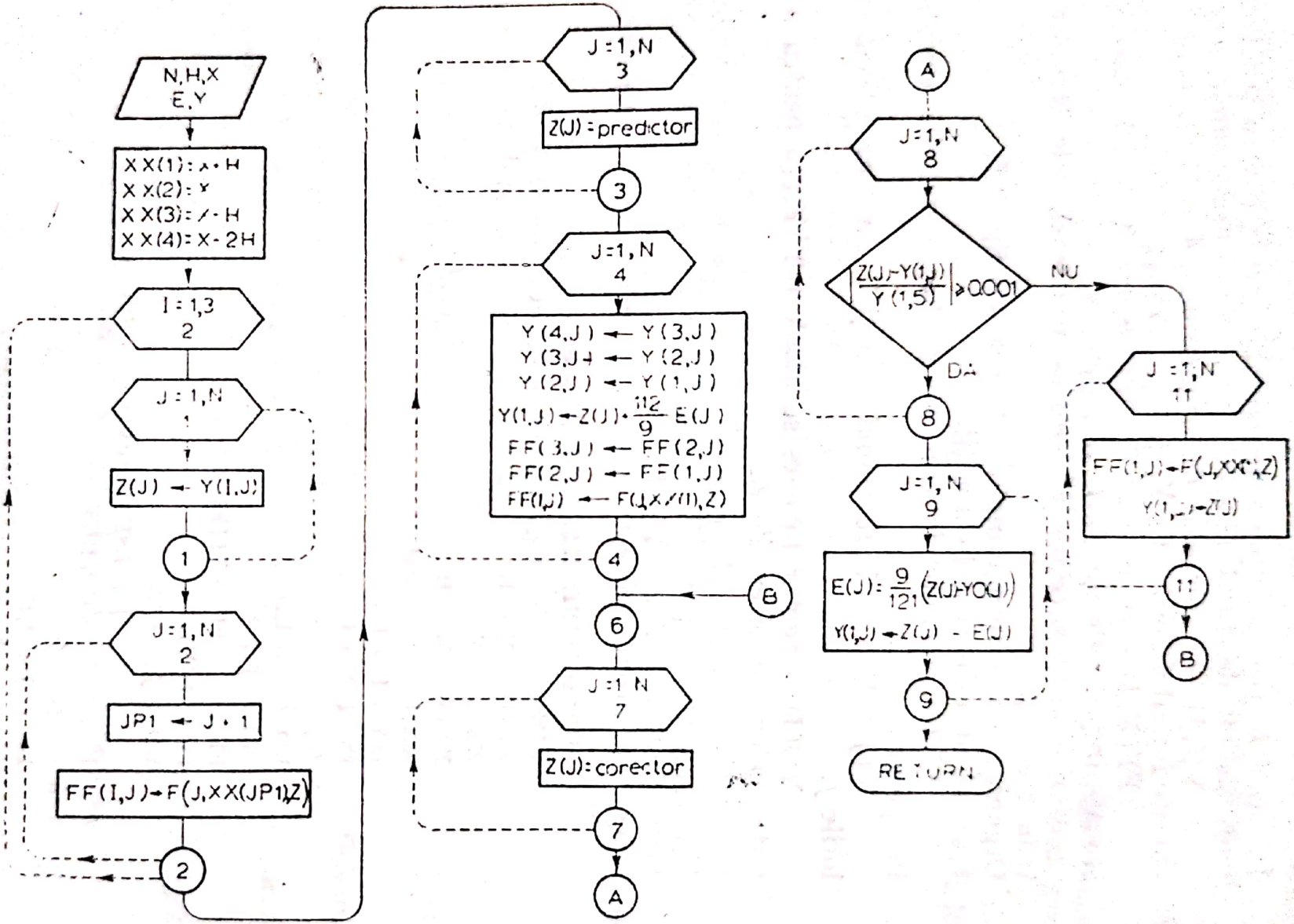


Fig. 2.5.

Scanned with OKEN Scanner

```
      Y(2,J) = Y(1,J)
      Y(1,J) = Z(J) + E(J)*112./9.
      Z(J) = Y(1,J)
      FF(3,J) = FF(2,J)
      FF(2,J) = FF(1,J)
    4 FF(1,J) = F(J,XX(1),Z)
    6 DØ 7 J = 1,N
    7 Z(J) = (9.*Y(2,J) - Y(3,J) + 3.*H*(FF(1,J) + 2.*FF(2,J)
     1        - FF(3,J)))/8.
      DØ 8 J = 1,N
      IF(ABS((Z(J) - Y(1,J))/Y(1,J)).LE.0.001)GØ TØ 10
    8 CØNTINUE
      DØ 9 J = 1,N
      E(J) = (Z(J) - YO(J))*9./121.
    9 Y(1,J) = Z(J) - E(J)
      RETURN
   10 DØ 11 J = 1,N
      FF(1,J) = F(J,XX(1),Z)
   11 Y(1,J) = Z(J)
      GØ TØ 6
      END
```

2.2.6. SIMULAREA REACȚIILOR CHIMICE

În cazul reacțiilor chimice cu mecanism complex sau cînd există mai multe reacții cuplate între ele, una dintre problemele deosebit de dificile este verificarea constantelor de viteză obținute experimental. În acest scop trebuie integrat sistemul de ecuații diferențiale care descrie evoluția în timp a reacțiilor.

De exemplu, sistemul de reacții

$$A \xrightarrow{k_1} B$$

$$2A \xrightarrow{k_2} C$$

este descris de trei ecuații diferențiale

$$\begin{aligned} \frac{d[A]}{dt} &= -(k_1[A] + k_2[A]^2) \\ \frac{d[B]}{dt} &= k_1[A] \\ \frac{d[C]}{dt} &= k_2[A]^2, \end{aligned} \tag{2.2.24}$$

Scanned with OKEN Scanner

unde mărimile înscrise între paranteze pătrate reprezintă concentraţiile componentelor respective, t este timpul şi k_1, k_2 sînt constantele de viteză.

Dacă sînt cunoscute concentraţiile iniţiale şi constantele de viteză, prin integrarea numerică a sistemului 2.2.24 se obţine variaţia în timp a concentraţiilor componentelor. Dacă valorile calculate reproduc datele experimentale, înseamnă că constantele de viteză sînt corecte, şi este corect şi mecanismul de reacţie pe baza căruia au fost calculate constantele şi formulat sistemul de ecuaţii diferenţiale.

Integrarea numerică se poate realiza cu una dintre metodele prezentate înainte, cu condiţia scrierii unui subprogram-funcţie care să corespundă sistemului de ecuaţii. Astfel, pentru sistemul (2.2.24) această funcţie ar fi:

```
      FUNCTIØN F(I,X,Y)
      DIMENSIØN Y(3), AK(2)
      DATA AK(1),AK(2)/....,..../
      IF(I-2)1,2,3
    1 F= - (AK(1) + AK(2)*Y(1))*Y(1)
      GØ TØ 4
    2 F = AK(1)*Y(1)
      GØ TØ 4
    3 F = AK(2)*Y(1)**2
    4 RETURN
      END
```

Argumentul X a fost introdus în lista de apel a funcţiei pentru ca ea să poată fi utilizată împreună cu programele de integrare descrise mai înainte. AK(I) şi Y(J) sînt constantele de viteză şi respectiv concentraţiile reactanţilor produşilor.
Considerăm în continuare un sistem de reacţie mai complex:

$$\begin{aligned} A + B &\xrightarrow{k_1} C + E \\ A + C &\xrightarrow{k_2} D + E \\ A + D &\xrightarrow{k_3} F + E \end{aligned} \qquad (2.2.25)$$

Un exemplu real de astfel de reacţie*) ar fi:

$$\begin{aligned} &C_6H_3(COOCH_3)_3 + {}^-OH \xrightarrow{k_1} (COOCH_3)_2 \cdot C_6H_3 \cdot COO^- + CH_3OH \\ &(COOCH_3)_2C_6H_3 \cdot COO^- + {}^-OH \xrightarrow{k_2} ({}^-OOC)_2C_6H_3 \cdot COOCH_3 + \\ &+ CH_3OH \\ &({}^-OOC)_2C_6H_3 \cdot COOCH_3 + {}^-OH \xrightarrow{k_3} C_6H_3({COO}^-)_3 + CH_3OH \end{aligned} \qquad (2.2.26)$$

*) **W. J. Svirbely, J. A. Blauer,** „J. Amer. Chem. Soc.", 83, 4118 (1961).

Scanned with OKEN Scanner

Sistemului de reacţii (2.2.25) îi corespunde următorul set de ecuaţii diferenţiale:

$$\frac{d[A]}{dt} = -(k_1[B] + k_2[C] + k_3[D])[A]$$

$$\frac{d[B]}{dt} = -k_1[B][A]$$

$$\frac{d[C]}{dt} = (k_1[B] - k_2[C])[A] \qquad (2.2.27)$$

$$\frac{d[D]}{dt} = (k_2[C] - k_3[D])[A]$$

$$\frac{d[F]}{dt} = k_3[D][A]$$

$$\frac{d[E]}{dt} = (k_1[B] + k_2[C] + k_3[D])[A]$$

Subprogramul-funcţie corespunzător acestui sistem de ecuaţii diferenţiale, în cazul particular al reacţiei de hidroliză a 1,3,5-tricarbometoxibenzenului este

```
      FUNCTIØN F(J,X,Y)
      DIMENSIØN Y(6), AK(3)
      DATA (AK(I), I = 1,3)/14.69,1.550,0.2899/
      IF(J-2) 1,2,3
    3 IF(J-4)4,5,6
    6 IF (J-5) 5,7,8
    1 F = -(AK(1)*Y(2) + AK(2)*Y(3) + AK(3)*Y(4))*Y(1)
      GØ TØ 9
    2 F=-AK(1)*Y(1)*Y(2)
      GØ TØ 9
    4 F = (AK(1)*Y(2)-AK(2)*Y(3))*Y(1)
      GØ TØ 9
    5 F = (AK(2)*Y(3)-AK(3)*Y(4))*Y(1)
      GØ TØ 9
    7 F = AK(3)*Y(4)*Y(1)
      GØ TØ 9
    8 F = (AK(1)*Y(2) + AK(2)*Y(3) + AK(3)*Y(4))*Y(1)
    9 RETURN
```

Constantele de viteză sînt exprimate în (l/mol.min). În continuare este listat programul principal care efectuează

Scanned with OKEN Scanner

simularea reacției de solvoliză a 1,3,5-tricarbometoxibenzenului, utilizînd subprogramul HEUN.

```
      DIMENSØN Y(6), Y1 (6), TITLU(20)
    1 READ(IN,100)TO,TF
      IF(TF)2,2,33
   33 READ(IN,103) GATA
      GØ TØ 3
    2 STØP
    3 READ(IN,101) (TITLU(J),J = 1,20)
      IF(TITLU(1).EQ.GATA) GØ TØ 4
      WRITE(IØ,200) (TITLU(J), J=1,20)
      GØ TØ 3
    4 READ(IN,102) (Y(I),I = 1,6)
```

Această parte a programului permite introducerea datelor necesare calculului propriu-zis:

TO, TF — limitele intervalului de integrare;
Y(I) — concentrațiile inițiale ale componentelor;
H — pasul de integrare inițial

și de asemenea face posibilă introducerea unor comentarii prin intermediul mărimii denumite TITLU (J).
Formatele folosite vor fi:

```
100 FØRMAT(2F10.5)
101 FØRMAT(20A4)
102 FØRMAT(7F10.5)
103 FØRMAT(A4)
200 FØRMAT(5X,20A4,/)
```

Un exemplu de cartele de comentarii ar fi:

```
ALFA
      HIDROLIZA TRICARBOXIMETILBENZENULUI — 11.03.1972
      SISTEMUL DE REACȚII ESTE
      A + B = C + E          K1 = 14. 69 L/MOL.MIN
      A + C = D + E          K2 = 1.550 L/MOL.MIN
      A + D = F + E          K3 = 0.2899 L/MOL.MIN
ALFA
```

În continuare programul conține următoarele instrucțiuni

```
      WRITE(IØ,201)H
      T = TØ
    7 WRITE(IØ,202)T, (Y(I),I = 1,6)
      CALL HEUN(6,H,T,Y,Y1)
      DØ 8 I = 1,N
```


Scanned with OKEN Scanner

```
8 Y(I) = Y1(I)
  T = T + H
  IF(T.GT.TF)GØ TØ 9
  GØ TØ 7
9 WRITE (IØ,203)
  GØ TØ 1
  END
```

Formatele utilizate sînt

```
201 FØRMAT (3X,'INTERVAL DE TIMP', 2X,F5.2,1X,'MINUTE'//
    1 2X,'TIMP', 7X, 'CØNCENTRAŢII PT. A,B,C,D,F,E'
    2 'MØL/LITRU'//)
202 FØRMAT (1X,F4.1,6 (2X,E12.5))
203 FØRMAT (//,' SFIRSIT' 1H1)
```

Această a doua parte a programului nu face decît să scrie datele transmise de subprogramul HEUN şi să prepare datele care trebuiesc transferate acestuia.

Singura secvenţă demnă de remarcat din acest program este modul de introducere a comentariilor: prin instrucţiunea 33 se citeşte mărimea GATA, în specificaţie alfanumerică (formatul 103) şi apoi pe măsură ce se citesc cartelele de comentariu sub forma vectorului TITLU(I), se compară TITLU(1) (adică primele patru coloane ale cartelei de comentariu) cu GATA. Dacă se întîlneşte o cartelă pentru care TITLU(1) este identic cu GATA, se întrerupe ciclul de citire a cartelelor de comentariu, trecîndu-se la instrucţiunea 4. În exemplul de cartele de comentariu, lui GATA i s-a dat conţinutul ALFA.

2.3. METODE STATISTICE

În multe ramuri ale ştiinţei şi industriei principale metodă de investigare este măsurarea parametrilor care caracterizează fenomenul studiat. Aceste măsurători prezintă două limite: metodele experimentale pot fi inexacte şi numărul determinărilor este deseori mic. În asemenea cazuri, folosind metodele statistice se poate obţine mai uşor interpretarea cea mai corectă a datelor. Aceaste metode permit experimentatorului proiectarea unei metode de rezolvare care să minimizeze erorile experimentale şi în a celaşi timp îi permit să evalueze siguranţa, valabilitatea rezultatelor.

Scanned with OKEN Scanner

În cazul unui studiu statistic, apar de obicei următoarele etape:

1. Alegerea unui model matematic care să reprezinte situaţia de fapt. Aceasta înseamnă:

— alegerea unei funcţii de distribuţie a datelor, bazat pe experienţă sau prin analogie;

— identificarea unor parametri ai distribuţiei cu date pe care vrem să le cunoaştem despre situaţia de fapt.

2. Alegerea unui sistem de luare a probelor.

3. Alegerea parametrilor statistici, calculaţi pe baza probelor, care vor fi folosiţi pentru formularea unor concluzii privind fenomenul studiat.

4. Formularea concluziilor:

— o estimare a parametrilor funcţiei de distribuţie presupuse pentru date împreună cu intervalele de încredere pentru aceşti parametri;

— o confirmare a valabilităţii modelului matematic luat în considerare iniţial.

De exemplu, în cazul multor procese tehnologice pentru proiectarea reactoarelor este suficientă cunoaşterea cineticii aparente a reacţiei chimice care are loc. În acest scop, se presupun pentru reacţie diverse ordine simple (reacţie unimoleculară sau bimoleculară etc.), verificîndu-se apoi care dintre ecuaţiile cinetice reproduce mai bine datele experimentale. În acest caz etapele generale formulate înainte vor deveni:

1. Scrierea ecuaţiei de viteză de reacţie corespunzătoare ordinului de reacţie ales şi exprimarea constantei de viteză funcţie de parametrul cunoscut experimental (de exemplu: concentraţia unei componente determinabile, intensitatea unei benzi în spectrul RMN, coeficientul de extincţie în cazul unui spectru UV etc.).

2. Se vor lua probe (analize cromatografice, spectre etc.) la intervale egale de timp.

3. Ca parametru statistic se va folosi abaterea medie patratică a valorilor calculate, cu ajutorul constantei de viteză deduse experimental, faţă de valorile experimentale al parametrului considerat (concentraţie, intensitate la spectre RMN etc.).

4. Se va considera corect acel ordin de reacţie pentru care s-a obţinut cea mai mică abatere.

Scanned with OKEN Scanner

O colecție de obiecte sau fenomene care au comun o anumită caracteristică observabilă sau măsurabilă, numită variat, formează o populație. Membrii populației se numesc indivizi, iar un grup de indivizi aleși la întîmplare din populație formează o probă.

O variabilă aleatoare este o mărime numerică asociată cu variatul, valoarea ei depinzînd de valoarea variatului pentru fiecare individ.

Probabilitatea (Pr) ca o variabilă aleatoare x să aibă valoarea A, pentru o anumită probă extrasă dintr-o populație, este egală cu fracția de indivizi din populație pentru care variabila X are valoarea A.

Funcția de frecventă sau funcția de densitate de probabilitate ($f(X_0)$. dX) reprezintă probabilitatea ca variabila aleatoare X să aibă valoarea cuprinsă între X_0 și $X_0 + dX$. Adică

$$f(X_0).dX = Pr\,(X_0 \leqslant X \leqslant X_0 + dX). \tag{2.3.1}$$

Ea are proprietatea:

$$\int_{-\infty}^{+\infty} f(X)\ dX = Pr\,(-\infty < X < +\infty) = 1. \tag{2.3.2}$$

Funcția de frecvență cumulativă sau funcția de distribuție este definită prin:

$$F(X) = Pr(-\infty < X \leqslant X_0) = \int_{-\infty}^{x_0} f(X)\ dX. \tag{2.3.3}$$

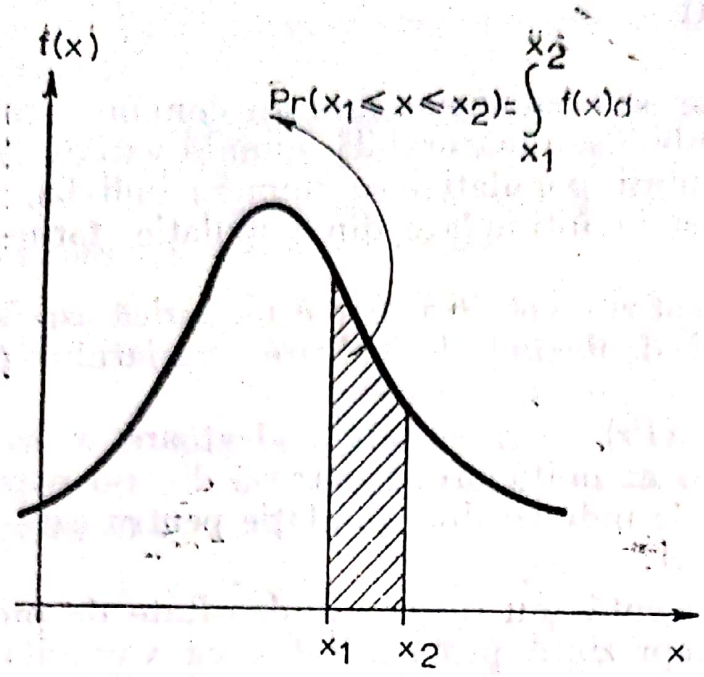
Ea reprezintă probabilitatea ca valoarea lui X să nu depășească X_0 și are următoarele proprietăți:

$$\begin{aligned} &F(-\infty) = 0; \\ &F(+\infty) = 1. \end{aligned} \tag{2.3.4}$$

Valoarea *așteptată* a unei variabile Y, care este funcție de variabila aleatoare X, se notează $E\,(Y)$ și se definește prin:

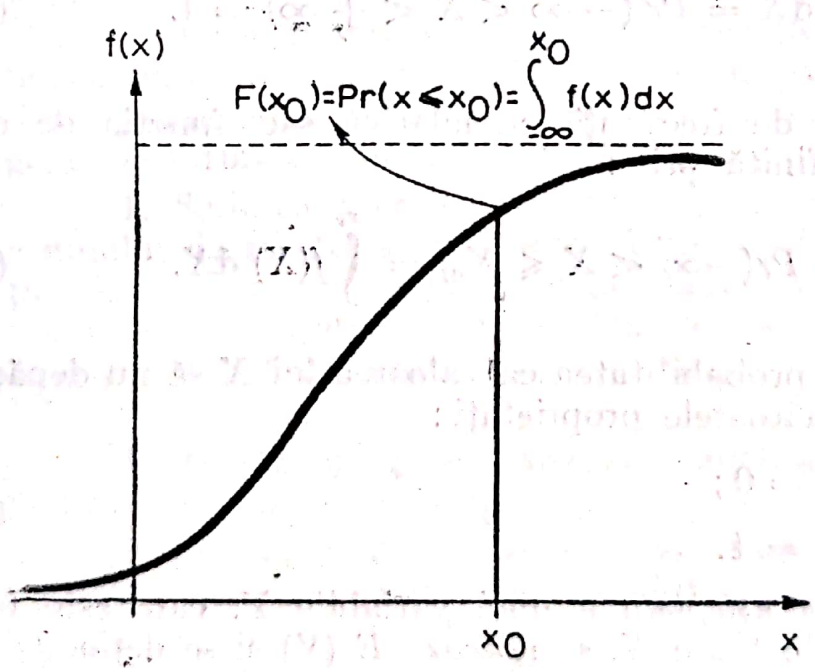
$$E(Y) = \int_{-\infty}^{+\infty} Y(X)\, f(X)\ dX. \tag{2.3.5}$$

Scanned with OKEN Scanner



funcția de frecvență

(a)



funcția de frecvență cumulată

(b)

Fig. 2.6.

Scanned with OKEN Scanner

Dacă ne referim la o distribuţie de frecvenţă particulară, atunci numim:

mod — valoarea cea mai frecventă a lui X,

medie (μ) — $\mu = E(X) = \int_{-\infty}^{+\infty} Xf(X)\,dX$;

mediană (μ_e) — valoarea lui x pentru care $F(X) = 1/2$.

Aceste trei mărimi sînt identice pentru distribuţii simetrice. Două mărimi des utilizate sînt momentele de ordinul r, definite prin:

$$\mu_r' = E(x^r);$$
$$\mu_r = E[(X - \mu)^r]. \tag{2.3.6}$$

Atunci

$$\mu = \mu_1' = E(X);$$
$$\mu_2 = E[(X - \mu)^2] = var\,(X) = \sigma^2. \tag{2.3.7}$$

Ultima relaţie reprezintă definiţia variantei ($var\,(X)$) şi a abaterii standard (σ).

2.3.2. PARAMETRII STATISTICI AI UNEI PROBE

Considerăm o probă care conţine n măsurări independente ale variabilei aleatoare x:

1. Media probei este:

$$\overline{X} = \frac{1}{n} \Sigma X_i. \tag{2.3.8}$$

2. Varianţa probei (dispersia) va fi:

$$S^2 = \frac{\Sigma(X_i - \overline{X})^2}{n - 1}. \tag{2.3.9}$$

Se poate demonstra că aceste mărimi sînt estimări ale mediei (μ) şi abaterii standard (σ) ale întregii populaţii. În cazul în care avem de-a face cu două variabile aleatoare x_i şi x_j se poate defini şi calcula o mărime, numită covarianţă, care exprimă gradul de independenţă al celor două variabile:

$$cov(x_i, x_j) = E[(x_i - \mu_{x_i})(x_j - \mu_{x_j})] = E(x_i, x_j) - \mu_{x_i} - \mu_{x_j}.$$

Scanned with OKEN Scanner

În cazul unei probe compuse din n determinări independente ale celor două variabile

$$cov(x_i, x_j) = \frac{1}{n-1}\left[\sum_{k=1}^{n} x_{ik}\, x_{jk} - \frac{\left(\sum_{k=1}^{n} x_{ik}\right)\left(\sum_{k=1}^{n} x_{jk}\right)}{n}\right]. \qquad (2.3.10)$$

O mărime derivată din covariantă este coeficientul de corelație

$$r_{ij} = \frac{cov(x_i, x_j)}{\sqrt{S^2_{x_i}\, S^2_{x_j}}}. \qquad (2.3.11)$$

2.3.3. PROGRAM PENTRU CALCULUL PARAMETRILOR STATISTICI AI UNEI PROBE

Pentru scrierea acestui program s-a folosit o instrucțiune FØRTRAN care nu a fost prezentată în capitolul 1. Această instrucțiune, denumită ENTRY, permite scrierea mai multor subprograme care operează cu un același set de mărimi sub forma unei subrutine unice cu mai multe puncte de intrare. În cazul nostru, mărimile comune vor fi doi vectori X1(I) și X2(J), care corespund valorilor a două variabile aleatoare pentru care vrem să evaluăm parametrii statistici.

S-au folosit următoarele notații:

AMIU — numele subprogramului de calcul a mediei;
DISP — numele subprogramului de calcul a dispersiei;
CØVR — subprogramul de calcul a covarianței;
CCØRL — subprogramul de calcul a coeficientului de corelație;
AMED — media;
S2 — dispersia;
CIJ — covarianța;
RIJ — coeficientul de corelație;
N — numărul de valori experimentale.

Subprogramul este compus din următoarele instrucțiuni:

```
      SUBRØUTINE AMIU(N,X1,AMED)
      DIMENSIØN X1(30), X2(30)
      AMED = 0.0
      DØ 1 I = 1,N
    1 AMED = AMED + X1(I)
      AMED = AMED/FLØAT(N)
```


Scanned with OKEN Scanner

```
      RETURN
      ENTRY DISP (N,X1,AMED,S2)
      S2 = 0.0
      DØ 2 I = 1,N
    2 S2 = S2 + (X1(I) - AMED)**2
      S2 = S2/FLØAT (N - 1)
      RETURN
      ENTRY CØVR(N,X1,X2, CIJ)
      A1 = 0.0
      A2 = 0.0
      A12 = 0.0
      DØ 3 I = 1,N
      A1 = A1 + X1(I)
      A2 = A2 + X2(I)
    3 A12 = A12 + X1(I)*X2(I)
      CIJ = (A12 + A1*A2/FLØAT(N))/FLØAT(N-1)
      RETURN
      ENTRY CCØRL(CIJ,S2I,S2J, RIJ)
      RIJ = CIJ/SQRT(S2I*S2J)
      RETURN
      END
```

În ultima parte a programului s-au notat cu S2J şi S2I dispersiile celor două variabile pentru care se calculează coeficientul de corelaţie.

Pentru utilizarea acestui subprogram se vor folosi următoarele secvenţe de chemare:

CALL AMIU (N,X1, AMED) — pentru calculul mediei;

CALL DISP (N,X1, AMED, S2) — pentru calculul dispersiei;

CALL CØVR (N,X1, X2, CIJ) — pentru calculul covarianţei;

CALL CCØRL (CIJ, S2I, S2J, RIJ) — pentru calculul coeficientului de corelaţie.

Această subrutină va fi folosită în cadrul programului de regresie multiliniară.

2.3.4. REGRESIE MULTILINIARĂ

Fenomenele studiate în chimie depind de cele mai multe ori de mai mulţi factori, care variază independent. De exemplu: randamentul unei reacţii poate să depindă de temperatură, presiune, concentraţie şi tip de catalizator, umiditate, amestecare etc.

Scanned with OKEN Scanner

Pentru a descrie efectele acestor factori asupra mărimii studiate, în foarte multe cazuri se folosesc ecuaţii liniare, de forma :

$$y = \sum_{i=1}^{n} a_i x_i + a_0. \tag{2.3.12}$$

Aceste ecuaţii nu au în general un conţinut fizic, în sensul că nu exprimă mecanismul fizic al dependenţelor ce apar. Ele sînt folosite pentru a descrie cantitativ datele experimentale, în scopul determinării unor condiţii în care mărimea studiată să aibă valori extreme. De exemplu, determinarea condiţiilor în care conversia este maximă sau randamentul într-un produs secundar neutilizabil este minim.

O ecuaţie de tipul (2.3.12) se numeşte ecuaţie de regresie multiliniară.

Dacă dispunem de un număr oarecare de date experimentale, putem determina coeficienţii a_i, folosind *metoda celor mai mici pătrate*, în felul descris în continuare. Suma abaterilor pătratice ale datelor experimentale în jurul ecuaţiei de regresie este :

$$\varepsilon = \sum_{k=1}^{m} \left(y_k - \left(a_0 + \sum_{i=1}^{n} a_i x_{ik} \right) \right)^2, \tag{2.3.13}$$

unde

m — numărul de date experimentale ;
n — numărul de variabile independente x_i ;
y — variabila dependentă.

Conform metodei celor mai mici pătrate, punem condiţia ca valorile coeficienţilor a_i să fie astfel alese, încît ε să aibă valoarea minimă, adică :

$$\frac{\partial \varepsilon}{\partial a_i} = 0, \quad i = 0,1,2,3 \ldots \; n. \tag{2.3.14}$$

Derivînd ecuaţia (2.3.13) obţinem un sistem de $(n + 1)$ ecuaţii cu $(n + 1)$ necunoscute :

$$\begin{aligned}
& m a_0 + a_1 \sum_k x_{1k} + \ldots + a_n \sum_k x_{nk} = \sum_k y_k \\
& a_0 \sum_k x_{1k} + a_1 \sum_k x_{1k} x_{1k} + \ldots + a_n \sum_k x_{1k} x_{nk} = \sum_k x_{1k} y_k \\
& \vdots \qquad\qquad \vdots \qquad\qquad\qquad\qquad \vdots \qquad\qquad \vdots \\
& a_0 \sum_k x_{nk} + a_1 \sum_k x_{nk} x_{1k} + \ldots + a_n \sum_k x_{nk} x_{nk} = \sum_k x_{nk} y_k.
\end{aligned} \tag{2.3.15}$$

Scanned with OKEN Scanner

Din prima ecuaţie putem scrie:

$$a_0 = \frac{\sum_{k=1}^{m} y_k}{m} - \sum_{i=1}^{n} \frac{a_i \sum_{k=1}^{m} x_{ik}}{m}. \tag{2.3.16}$$

Introducînd într-una din ecuaţiile următoare obţinem:

$$\sum_{j=1}^{n} a_j \left(\sum_{k=1}^{m} x_{ik} x_{jk} - \frac{\left(\sum_{n=1}^{m} x_{ik}\right)\left(\sum_{k=i}^{m} x_{jk}\right)}{m} \right) = \tag{2.3.17}$$

$$= \sum_{k=1}^{m} x_{ik} y_k - \frac{\left(\sum_{k=1}^{m} x_{ik}\right)\left(\sum_{k=1}^{m} y_k\right)}{m}.$$

Împărţind această expresie cu $(m-1)$ se obţine:

$$\sum_{j=1}^{n} a_j \, . \, cov(x_i, x_j) = cov(x_i, y). \tag{2.3.18}$$

Împărţind acum prin $S_{x_i}^2 \cdot S_y^2$ obţinem:

$$\sum_{j=1}^{n} a_j \frac{cov(x_i, x_j)}{\sqrt{S_{x_i}^2 \, S_y^2}} = \frac{cov(x_i, y)}{\sqrt{S_{x_i}^2 \, S_y^2}}. \tag{2.3.19}$$

Dacă se definesc drept coeficienţi de regresie standard:

$$B_j = \left(a_j \sqrt{S_{x_j}^2}\right) / \sqrt{S_y^2} \tag{2.3.20}$$

şi ştiind că:

$$r_{ij} = \frac{cov(x_i, x_j)}{\sqrt{S_{x_i}^2 \, S_{x_j}^2}}$$

ecuaţia (2.3.19) devine:

$$\sum_{j=1}^{n} B_j r_{ij} = r_{yi}. \tag{2.3.21}$$

Obţinem astfel un nou sistem de n ecuaţii cu n necunoscute numit *sistem de ecuaţii normale standardizate*. Dacă variabilele

x_i sînt independente, $r_{ii} = 1$, $r_{ij} = r_{ji}$ și atunci sistemul este:

$$\begin{aligned} &B_1 + B_2 r_{12} + \ldots + B_n r_{1n} = r_{y1} \\ &B_1 r_{12} + B_2 + \ldots + B_n r_{2n} = r_{y2} \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &B_1 r_{1n} + B_2 r_{2n} + \ldots + B_n = r_{yn}. \end{aligned} \tag{2.3.22}$$

Deci, pentru obținerea coeficienților a_i se rezolvă întîi sistemul 2.3.22, se calculează apoi a_i pentru $i = 1,2 \ldots n$ cu ecuația 2.3.20 și la urmă se calculează a_0 cu ecuația 2.3.16.

2.3.5. PROGRAM PENTRU REGRESIE MULTILINIARĂ

Pentru scrierea unui program de regresie multiliniară vom lua în considerare următorul algoritm:

1. Citește valorile variabilei dependente y.
2. Citește valorile variabilelor independente x_i. Pe măsură ce se citesc valorile corespunzătoare unei variabile x_i se calculează r_{yi} și se compară această valoare cu o valoare r scoasă din tabelul 2.1.

Dacă valoarea $r_{yi} > r_v$ înseamnă că variabilele y și x_i cu o probabilitate de 95% nu sînt independente.

TABELUL 2.1

Grade de libertate $=m-2$	r_v		
1	0,397	15	0,482
2	0,950	16	0,468
3	0,878	17	0,456
4	0,811	18	0,444
5	0,754	19	0,433
6	0,707	20	0,423
7	0,666	25	0,381
8	0,632	30	0,349
9	0,602	40	0,304
10	0,576	50	0,273
11	0,553	60	0,250
12	0,532	80	0,217
13	0,514	100	0,195
14	0,497		

Scanned with OKEN Scanner

Vor fi reţinute numai acele variabile x_i pentru care este îndeplinită condiţia precedentă. Valoarea r_v va trebui să figureze printre datele iniţiale ale programului.

3. Se construieşte sistemul de ecuaţii normale standardizate.

4. Se rezolvă sistemul folosind subprogramul de eliminare Gauss-Jordan prezentat în capitolul 2.4.

5. Se calculează coeficienţii a_i.

6. Se calculează dispersia valorilor experimentale în jurul ecuaţiei de regresie

$$S^2_{y,\ x_1,\ x_2,\ldots x_n} = \frac{\sum_{i=1}^{m} (y_i - y_{i,calc.})^2}{m-(n+1)},$$

unde:

$y_{i,\ calc}$ — valori calculate, cu ecuaţia de regresie;

n — numărul de variabile independente luate în considerare la calculul de regresie;

m — numărul de puncte experimentale.

7. Se calculează coeficientul de corelaţie total

$$R = \sqrt{1 - \frac{S^2_{y,\ x_1,\ x_2,\ldots x_n}}{S^2_y}}$$

care exprimă măsura în care y depinde de toate variabilele x_i. Cu cît R este mai aproape de 1 înseamnă că variaţia lui y provine numai din variaţiile variabilelor x_i.

Programul va fi compus din următoarele instrucţiuni:

```
      DIMENSIØN Y(50), X(10,50), SX2(10), RIJ(10,10),
     * RYI(10), B(10), A(11), TITLU(20), X1(50), X2(50)
    1 READ(IN,100) NPUNCT, RNIU, NVI
      IF(NPUNCT) 99,99,2
    2 READ(IN,101) (TITLU(I), I = 1,20)
      WRITE (IØ,200) (TITLU(I), I = 1,20)
      II = 1
    3 IU = II + 7
      READ(IN, 102) (Y(I), I = II, IU)
      II = IU + 1
      IF(II-NPUNCT)3,3,4
```

```
 4 IVI = 1
   NEC = NVI
   CALL AMIU(NPUNCT, Y, YMED)
   CALL DISP(NPUNCT, Y, YMED, SY2)
 5 II = 1
 6 IU = II + 7
   READ(IN,102)(X(IVI,J), J = II, IU)
   II = IU + 1
   IF(II-NPUNCT)6,6,7
 7 DØ 8 J = 1,NPUNCT
 8 X1 (J) = X (IVI,J)
   CALL AMIU(NPUNCT,X1,XMED)
   CALL DISP(NPUNCT,X1,XMED,SX2 (IVI))
   CALL CØVR(NPUNCT, X1, Y,CYJ)
   CALL CCØRL (CYI, SX2(IVI), SY2, RYI (IVI))
   IF(RYI (IVI)-RNIU)9,90,10
 9 NEC = NEC - 1
   IF(NEC - IVI)11,5,5
10 IVI = IVI + 1
   IF(NEC - IVI)11,5,5
11 CØNTINUE
```

Instrucţiunile listate pînă acum corespund primelor două etape ale algoritmului de calcul. S-au folosit următoarele notaţii:

NPUNCT—număr de puncte experimentale;

RNIU — r_v;

NVI — număr de variabile independente luate în considerare iniţial;

Y(I) — variabila dependentă;

X(I, J) — variabilele independente;

RIJ(I, J) — r_{ij};

RYI(I) — r_{yi};

B(I) — B_i;

A(J) — a_i;

X_1(I), X2(I) — doi vectori care sînt folosiţi pentru stocarea temporară a valorilor x_{ik} pentru a putea fi transmise subrutinei de calcul a dispersiei, covarianţei şi coeficientului de corelaţie;

NEC — numărul de ecuaţii;

IVI — indicele variabilei independente;

S2Y — S_y^2;

S2X(I) — $S_{x_i}^2$;

Scanned with OKEN Scanner

XMED — mărime în care se stochează temporar mediile variabilelor independente ;

YMED — valoarea medie a variabilei dependente y.

Pentru stabilirea numărului de ecuaţii, adică a numărului de variabile independente care vor fi luate în considerare, se procedează în felul următor :

— se iniţializează NEC = NVI şi IVI = 1 ;

— se citesc valorile X(IVI, J) cu J = 1, NPUNCT şi se transferă prin vectorul X1(J) subrutinele AMIU, DISP, CØVR şi CCØRL, obţinîndu-se astfel RYI (IVI) ;

— dacă RYI (IVI) este mai mare decît RNIU atunci se măreşte IVI cu 1 şi se trece la citirea unui nou set de date pentru X(IVI, J) ; dacă RYI (IVI) este mai mic decît RNIU înseamnă că variabila independentă respectivă nu are efect asupra lui y şi deci poate fi ştearsă din memorie. Pentru aceasta valoarea lui IVI se lasă neschimbată şi în acest caz noile valori care vor fi citite se înscriu în memorie peste valorile x_{ik} citite înainte. De asemenea, numărul de ecuaţii va fi mai mic cu 1 şi deci NEC = = NEC—1 ;

— citirea de date se întrerupe după ce IVI a egalat valoarea lui NEC.

Conform algoritmului, următoarea etapă de calcul constă în obţinerea coeficienţilor RIJ (I, J). Aceasta se realizează prin următorul set de instrucţiuni :

```
      DØ 13 J = 1,NEC
      DØ 13 K = J,NEC
      DØ 12 I = 1,NPUNCT
      X1(I) = X(J,I)
   12 X2(I) = X(K,I)
      CALL CØVR(NPUNCT,X1,X2,CIJ)
      CALL CCØRL(CIJ,SX2(J),SX2(K),RIJ(J,K))
      RIJ (K,J) = RIJ (J,K)
   13 CØNTINUE
```

La scrierea acestei zone a programului s-a ţinut cont de faptul că valorile SX2(I) au fost deja calculate în prima parte a programului.

Acum trebuie chemată subrutina de rezolvat sisteme de ecuaţii :

```
      CALL GAUJØR(NEC, 1, RIJ, RYI, B)
```


Scanned with OKEN Scanner

Obţinem astfel valorile B(I). În continuare programul conţine următoarele instrucţiuni:

```
      DØ 14 I = 1,NEC
      IP1 = I + 1
   14 A(IP1) = B(I)*SY2/SX2(I)
      A(1) = 0.0
      DØ 15 J = 1,NEC
      JP1 = J + 1
      DØ 15 K = 1,NPUNCT
   15 A(1) = A(1) - X(J,K)*A(JP1)
      DØ 17 J = 1,NPUNCT
   17 A(1) = A(1) + Y(J)
      A(1) = A(1)/FLØAT(NPUNCT)
```

Cu ajutorul acestei secvenţe de program s-au obţinut coeficienţii $a_i = A(I + 1)$. Deci a_0 este stocat în A(1).

În continuare urmează să calculăm valorile $y_{i,\,calc}$ şi apoi $S^2_{y,\,x_1,\,x_2,\,\ldots\,x_n}$ şi R. Valorile $y_{i,\,calc}$ le vom stoca în X1 iar dispersia valorilor experimentale în jurul ecuaţiei de regresie şi coeficientul de corelaţie total le vom nota cu SDR şi RT:

```
      DØ 18 J = 1,NPUNCT
      X1(J) = A(1)
      DØ 18 I = 1,NEC
      IP1 = I + 1
   18 X1(J) = X1(J) + A(JP1)*X(I,J)
      SDR = 0.0
      DØ 19 J = 1,NPUNCT
      X2(J) = Y(J) - X1(J)
   19 SDR = SDR + X2(J)**2
      SDR = SDR/FLØAT(NPUNCT-NEC + 1)
      R = SQRT(1.0-SDR/SY2)
```

Ca rezultate finale se vor scrie pe imprimantă:

- un tabel cu valorile x_{ik}, y_i, $y_{i,\,calc}$, $y_i - y_{i,\,calc}$;
- un tabel cu valorile r_{ij} şi r_{yi};
- valorile SDR şi R.

```
      WRITE(IØ,201)NPUNCT,RNIU,NEC
      DØ 20 I = 1,NPUNCT
   20 WRITE(IØ,202)Y(I),X1(I),X2(I),(X(J,I),J = 1,NEC)
      DØ 21 I = 1,NEC
```


Scanned with OKEN Scanner

```
 21 WRITE(IØ,203)RYI(I),(RIJ(I,J),J = 1,NEC)
    WRITE(IØ,204)SDR,R
    GØ TØ 1
 99 STØP
    END
```

Formatele utilizate de acest program sînt:

```
100 FØRMAT(I3,F10.5,13)
101 FØRMAT(20A4)
102 FØRMAT(8F10.5)
200 FØRMAT(1H1,20X,20A4,//)
201 FØRMAT(20X,'NR. DE PUNCTE',2X,I3,//,20X,'RNIU',
    1 10X,F10.5,///,5X,' Y ',8X,' Y CALC.',5X,
    2 'Y—YCALC',15X,'X(1),X(2),....X(',I2,')',//)
202 FØRMAT (1X,10(E11.4,1X))
203 FØRMAT (//,5X,'RYI',20X,'RIJ(I,J)',//
    1 1X,10(E11.4,1X))
204 FØRMAT (////,10X,'DISPERSIE SDR', 3X,E12.5,//
    1 ,'CØRELATIE RT',3X,E12.5)
```

2.4. CALCUL CU MATRICE

În acest capitol se vor prezenta o serie de noţiuni de bază privind matricele şi vectorii. Aceste noţiuni sînt necesare pentru rezolvarea unor probleme foarte des întîlnite în practică, ca soluţionarea sistemelor de ecuaţii liniare (problemă care apare şi în cadrul rezolvării ecuaţiilor diferenţiale cu derivate parţiale) sau obţinerea valorilor şi vectorilor proprii în cadrul metodelor de calcul a structurii electronice a moleculelor.

2.4.1. DEFINIŢII ŞI NOŢIUNI DE BAZĂ

Printr-o *matrice* înţelegem un tablou de $n \times m$ elemente, care pot fi numere reale sau complexe:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \ldots & a_{1m} \\ a_{21} & a_{22} & a_{23} \ldots & a_{2m} \\ \cdot & \cdot & \cdot \cdot \cdot & \cdot \\ a_{n1} & a_{n2} & a_{n3} \ldots & a_{nm} \end{bmatrix} \qquad (2.4.1)$$

Scanned with OKEN Scanner

O matrice care are toate elementele egale cu zero se numeşte *matrice nulă* şi se notează cu 0.

Adunarea şi scăderea matricelor se realizează prin adunarea şi respectiv scăderea elementelor corespunzătoare din cele două matrice. Astfel pentru

$$C = A + B$$

avem

$$c_{ij} = a_{ij} + b_{ij}. \tag{2.4.2}$$

Matricele A şi B trebuie să aibă aceleaşi dimensiuni.

Ca două matrice să poată fi înmulţite trebuie ca numărul de coloane al primei să fie egal cu numărul de linii al celei de a doua. Astfel, dacă A are dimensiunea $n \times m$ iar B dimensiunea $m \times q$, prin înmulţire se obţine matricea C, de dimensiune $n \times q$:

$$C = A \cdot B$$

$$c_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj}. \tag{2.4.3}$$

În cazul în care $n = q$ este posibil şi produsul BA. Dacă $AB = BA$, spunem că cele două matrice *comută*.

Matricea ale cărei elemente sînt numerele complex conjugate ale elementelor matricei A se notează cu $\bar{A}$ şi se numeşte *matricea complex conjugată* a lui A.

Prin *matricea transpusă* A^t înţelegem o matrice ale cărei elemente (r_{ij}) sînt legate de cele ale matricei A prin relaţia $r_{ij} = a_{ji}$. Prin transpunere, practic coloanele devin liniile noii matrice. Matricea transpusă are următoarele proprietăţi:

$$\begin{aligned} &B^t + C^t = (B + C)^t \\ &(AB)^t = A^t \cdot B^t. \end{aligned} \tag{2.4.4}$$

Matricea *complex conjugat transpusă* A^* a unei matrice A se defineşte prin:

$$A^* = (\bar{A^t}).$$

Două proprietăţi importante ale acestor matrice sînt:

$$\begin{aligned} &(\bar{A})^t = (\bar{A^t}) \\ &(A \cdot B)^* = A^* \cdot B^*. \end{aligned} \tag{2.4.5}$$

Spunem despre o matrice A că este *hermitică* dacă $A^* = A$. (Posibil cu condiția ca matricea să fie pătrată.) O matrice reală (ale cărei elemente sînt numere reale) este hermitică numai dacă este *simetrică*, adică $A = A^t$.

Vom spune despre o matrice pătrată R că este *diagonală* dacă $r_{ij} = 0$ pentru $i \neq j$. Dacă considerăm că R are dimensiunea $n \times n$ iar A și B dimensiunile $m \times n$ și, respectiv, $n \times q$ avem:

$$C = A \cdot R \qquad c_{ij} = r_{jj} a_{ij}$$

$$D = R \cdot B \qquad d_{ij} = r_{ii} b_{ij}.$$

Dacă $r_{ii} = r$, indiferent de i, matricea R se numește *matrice scalar*, iar dacă $r = 1$ obținem *matricea identitate* care se notează I și are proprietatea

$$AI = IA = A.$$

Cu ajutorul matricei identitate se poate defini *matricea inversă*, și anume: dacă A și K sînt două matrice pătrate cu proprietatea $AK = I$, atunci K se numește matricea inversă a lui A și se notează A^{-1}. În această situație se spune că matricea A este *nesingulară*.

Fiecărei matrice pătrate i se asociază un număr numit *determinantul matricei* și notat $\det(A)$, $\det A$ sau A. Determinantul corespunzător matricei ce se obține scoțînd din matricea A linia i și coloana j este numit *minorul* corespunzător lui a_{ij}. Dacă acest minor se înmulțește cu $(-1)^{i+j}$ atunci obținem *cofactorul* lui a_{ij}. Notînd cofactorul cu A_{ij}, avem:

$$\det(A) = \sum_{r=1}^{m} a_{rj} A_{rj}, \tag{2.4.6}$$

unde r poate avea orice valoare n (dimensiunea matricei). Acest mod de a dezvolta un determinant după minori se numește dezvoltare Laplace.

Se numește *matrice adjunct* a lui A matricea formată din cofactorii elementelor sale. Este valabilă proprietatea:

$$A \cdot \text{adj}\,(A) = \det(A) \cdot I.$$

Dacă $\det(A) \neq 0$ se poate calcula ușor matricea inversă. Elementele ei vor fi

$$f_{ij} = A_{ji}/\det(A). \tag{2.4.7}$$

Scanned with OKEN Scanner

Acest procedeu de a inversa o matrice este însă laborios, în practică preferîndu-se tehnici mai complicate ca logică dar mai simplu de realizat pe calculator.

2.4.2. PROGRAME. OPERAȚII CU MATRICE PĂTRATE. TIPĂRIREA MATRICELOR

Se prezintă în continuare o subrutină cu mai multe puncte de intrare, care efectuează operațiile de adunarea matricelor, scăderea matricelor, transpunerea matricelor, înmulțirea matricelor și inversarea matricelor.

Primele trei operații se fac folosind formulele (2.4.2) și (2.4.3). În cazul înmulțirii, rezultatul va fi stocat într-una din matricele inițiale. La fel, în cazul transpunerii și inversării, rezultatul va apărea în matricea inițială. Organigrama secvenței de inversare este cea din fig. 2.7

Programul este compus din următoarele instrucțiuni :

```
      SUBRØUTINE ADMAT (N,A,B,C)
      DIMENSIØN A (30,30), B(30,30), C(30,30), X(30)
      DØ 1 J = 1,N
      DØ 1 K = 1,N
    1 C(J,K) = A(J,K) + B(J,K)
      RETURN
      ENTRY SCAMAT(N,A,B,C)
      DØ 2 J = 1,N
      DØ 2 K = 1,N
    2 C(J,K) = A(J,K) - B(J,K)
      RETURN
      ENTRY MMULT(A,B,N,C)
      DØ 3 J = 1,N
      DØ 4 K = 1,N
      X(K) = A (J,K)
    4 C(J,K) = 0.0
      DØ 3 K = 1,N
      DØ 3 I = 1,N
      C(J,K) = C(J,K) + X(I)*B(I,K)
    3 CØNTINUE
      RETURN
      ENTRY TRMAT (N,A)
      NM1 = N - 1
      DØ 5 I = 1, NM1
      IP1 = I + 1
```


Scanned with OKEN Scanner

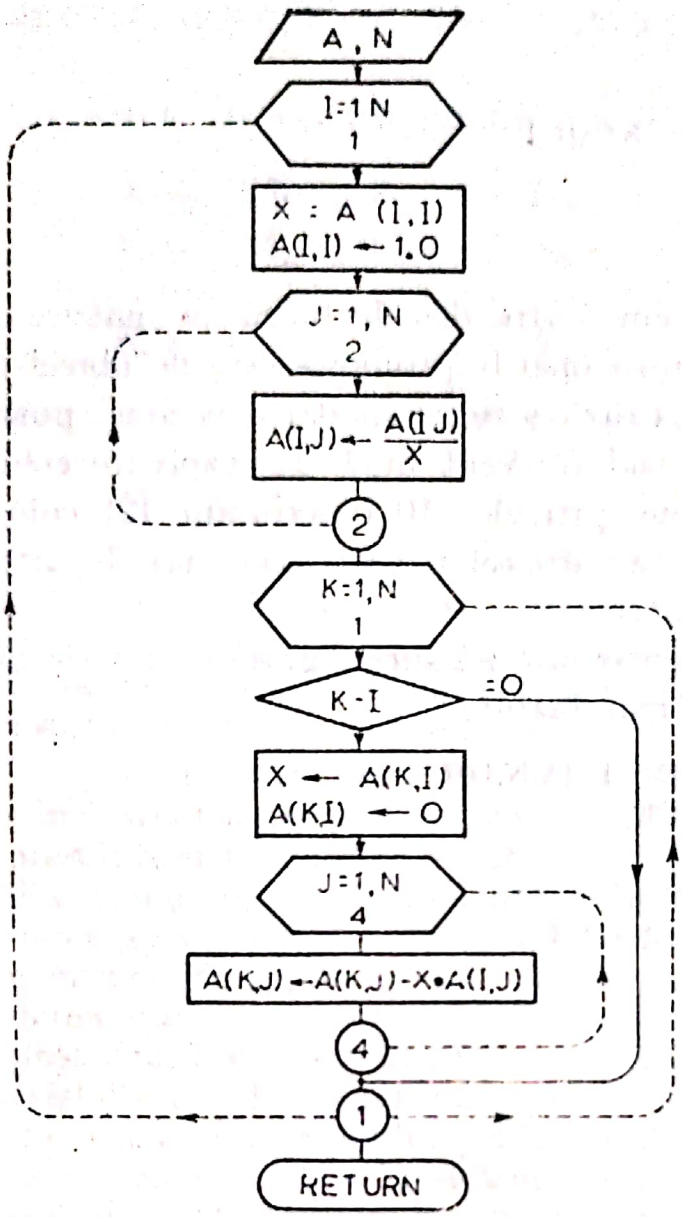


Fig. 2.7

```
      DØ 5 J = IP1,N
      X = A(I,J)
      A(I,J) = A(J,I)
5     A(J,I) = X
      RETURN
      ENTRY MATINV(A,N)
      DØ 9 I = 1,N
      X = A(I,I)
      DØ 6 J = 1,N
6     A(I,J) = A(I,J)/X
      DØ 9 K = 1,N
      IF(K-I)7,9,7
```


Scanned with OKEN Scanner

```
    X = A(K,I)
    A(K,I) = 0.0
    DØ 8 J = 1,N
8   A(K,J) = A(K,J) - X*A(I,J)
9   CØNTINUE
    RETURN
    END
```

În practică, avem foarte des de lucru cu matrice de dimensiuni mari. Lărgimea unei imprimante este de obicei de 132 coloane. Din această cauză scrierea matricelor mari pune probleme deosebite. Cel mai frecvent mod de rezolvare este acela de a scrie întîi termenii primelor 10 (maximum 12) coloane și apoi sub ei termenii următoarelor 10 și așa mai departe pînă la scrierea întregii matrice.

Se listează în continuare un subprogram care scrie matrice conform schemei de mai înainte:

```
    SUBRØUTINE TIMAT (A,N,IDIAG,IØ)
    DIMENSIØN A(30,30)
7   FØRMAT(1H0)
8   FØRMAT(I3,3X,12F8.4)
9   FØRMAT(1H0,2X,12 (6X,I2))
    IF(IDIAG)1,2,1
1   DØ 11 M = 2,N
    JP1 = M - 1
    DØ 11 L = 1,JP1
11  A(M,L) = A(L,M)
2   KITE = 0
20  LØWW = KITE + 1
    KITE = KITE + 12
    IF(KITE-N) 50,50,60
60  KITE = N
50  CØNTINUE
    WRITE (IØ,9) (I,I = LØWW, KITE)
    DØ 32 I = 1,N
32  WRITE (IØ,8)I, (A(I,J),J = LØWW,KITE)
    IF(N-KITE)40,40,20
40  WRITE (IØ,7)
    RETURN
    END
```


Scanned with OKEN Scanner

2.4.3. REZOLVAREA SISTEMELOR DE ECUAŢII LINIARE

Un sistem de ecuaţii liniare are forma:

$$\begin{aligned} b_{11}x_1 + b_{12}x_2 + \ldots + b_{1n}x_n &= u_1 \\ b_{21}x_1 + b_{22}x_2 + \ldots + b_{2n}x_n &= u_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots & \\ b_{n1}x_1 + b_{n2}x_2 + \ldots + b_{nn}x_n &= u_n \end{aligned} \tag{2.4.8}$$

Acest sistem se poate scrie în formă matricială;

$$Bx = u. \tag{2.4.9}$$

Pentru rezolvarea unui asemenea sistem, avem la dispoziţie două tipuri de metode: metode exacte şi metode de aproximări succesive. Primele sînt utile pentru sisteme cu mai puţin de 40 de ecuaţii, iar cele din urmă sînt mai convenabile pentru sisteme mari, pe care primele le rezolvă prea lent.

Una dintre metodele exacte folosite frecvent este metoda eliminării Gauss-Jordan. Metoda se bazează pe una dintre proprietăţile sistemelor de ecuaţii liniare, şi anume aceea că o ecuaţie din sistem poate fi înlocuită printr-una obţinută prin combinarea sa cu o altă ecuaţie din sistem. Prin combinare înţelegem aici înmulţirea cu un număr oarecare şi adunarea sau scăderea dintr-o altă ecuaţie a sistemului. Eliminarea Gauss-Jordan procedează în felul următor: folosind prima ecuaţie şi pe rînd celelalte $(n-1)$ din sistem, se elimină necunoscuta x_1 din acestea din urmă. De exemplu, pentru a elimina x_1 din a doua ecuaţie este suficient să o înmulţim cu b_{11} şi să o scădem din prima ecuaţie înmulţită cu b_{12}. Se repetă această operaţie eliminînd pe x_2 din toate ecuaţiile, exceptînd a doua. Continuîndu-se astfel, după n serii de eliminări, în fiecare ecuaţie va rămîne numai o necunoscută şi termenul liber. Impărţind termenul liber la coeficientul necunoscutei, obţinem valoarea acesteia.

Acest algoritm capătă o formă mult mai convenabilă dacă utilizăm forma matricială a sistemului.

De exemplu, considerăm sistemul:

$$\begin{aligned} 2x_1 - 7x_2 + 5x_3 &= 3 \\ x_1 + 9x_2 - 6x_3 &= 1 \\ -3x_1 + 8x_2 + 5x_3 &= 28 \end{aligned}$$

Formăm matricea

$$\begin{bmatrix} 2 & -7 & 5 & 3 \\ 1 & 9 & -6 & 1 \\ -3 & 8 & 5 & 28 \end{bmatrix}$$

Dăm factor comun în linia întîia pe termenul din prima coloană (termenul pivot) şi obţinem:

$$2 \times \begin{bmatrix} 1 & -7/2 & 5/2 & 3/2 \\ 1 & 9 & -6 & 1 \\ -3 & 8 & 5 & 28 \end{bmatrix}$$

Reducem la zero termenii din prima coloană ai celei de a doua şi a treia linii, scăzînd prima linie din a doua şi adunînd de trei ori prima linie la a treia. Obţinem:

$$2 \times \begin{bmatrix} 1 & -7/2 & 5/2 & 3/2 \\ 0 & 25/2 & -17/2 & -1/2 \\ 0 & -5/2 & 25/2 & 65/2 \end{bmatrix}$$

Repetăm operaţia pentru linia a doua: dăm factor comun pe 25/2 şi reducem termenii coloanei a doua din prima şi a treia linie:

$$2 \times 25/2 \times \begin{bmatrix} 1 & 0 & 3/25 & 34/25 \\ 0 & 1 & -17/25 & -1/25 \\ 0 & 0 & 54/5 & 162/5 \end{bmatrix}$$

Repetînd pentru linia a treia obţinem:

$$2 \times 25/2 \times 54/5 \times \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

Soluţiile sistemului sînt cifrele care apar în cea de a patra coloană, iar factorul comun este chiar valoarea determinantului format din coeficienţii ecuaţiei.

După cum se poate vedea, acest sistem permite rezolvarea simultană a mai multor sisteme de ecuaţii care diferă între ele numai prin termenii liberi. Pentru aceasta este suficient să se completeze matricea de lucru cu coloane suplimentare conţinînd diversele seturi de termeni liberi.

Scanned with OKEN Scanner

Metodele de rezolvare iterativă se bazează pe o reformulare a sistemului de ecuaţii 2.4.8, şi anume prin explicitarea din fiecare ecuaţie a uneia dintre necunoscute: din prima ecuaţie prima necunoscută, din a doua a doua necunoscută etc.

Astfel obţinem:

$$\begin{aligned} x_1 &= (u_1 - b_{12}x_2 - b_{13}x_3 - \ldots - b_{1n}x_n)/b_{11} \\ x_2 &= (u_2 - b_{21}x_1 - b_{23}x_3 - \ldots - b_{2n}x_n)/b_{22} \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n &= (u_n - b_{n1}x_1 - b_{n2}x_2 - \ldots - b_{n,n-1}x_{n-1})/b_{nn} \end{aligned} \qquad (2.4.10)$$

sau, mai concis:

$$x_i = \frac{\left(u_i - \sum_{j \neq i} b_{ij}x_j\right)}{b_{ii}}; \quad i = 1,2, \ldots n. \qquad (2.4.11)$$

Această rearanjare este posibilă doar dacă $b_{ii} \neq 0$. Pentru aceasta deseori ordinea necunoscutelor trebuie schimbată pentru a se îndeplini condiţia.

În continuare se alege un set oarecare de valori iniţiale pentru necunoscute. Acest set se introduce în partea dreaptă a ecuaţiilor (2.4.10), calculîndu-se astfel un nou set de valori pentru necunoscute. Noul set se reintroduce în calcul. Acest mod de lucru se numeşte metoda iterativă JACOBI.

O a doua metodă de acelaşi fel se deosebeşte de aceasta prin aceea că nu se aşteaptă ca setul de necunoscute să fie calculat în întregime, ci pe măsură ce o necunoscută s-a calculat, ea se introduce în calculul celor care urmează. Astfel setul iniţial este utilizat numai pentru calculul primei necunoscute. Această metodă poartă numele de metoda iterativă Gauss-Siedel. Criteriul de convergenţă pentru ambele metode este:

$$|x_{i,k+1} - x_{i,k}| < \varepsilon \qquad i = 1,2,\ldots,n,$$

adică modificarea obţinută după o iteraţie să fie pentru toate necunoscutele sub o anumită limită.

2.4.4. PROGRAME. METODA GAUSS-JORDAN. METODA GAUSS-SIEDEL

Algoritmul unui subprogram care utilizează metoda eliminării Gauss-Jordan pentru rezolvarea unui set de sisteme de ecuaţii liniare, care diferă între ele numai prin valorile termenilor liberi, este următorul:

Scanned with OKEN Scanner

— primește de la programul principal matricea coeficienților (B) și matricea termenilor liberi (U);
— construiește matricea mărită $A = [B : U]$;
— inițializează valoarea determinantului sistemului $d = 1$;
— calculează pentru $k = 1,2 \ldots n$ (unde n este numărul de ecuații din sistem):
valoarea determinantului $d = d \times a_{kk}$;
verifică valoarea elementului pivot; dacă este prea mic pot apărea erori deosebite și atunci renunță la calcul;
normarea

$$a_{kj} = a_{kj}/a_{kk},\ j = n + m,\ n + m - 1,\ \ldots,\ k,$$

unde m este numărul de seturi de termeni liberi. Această operație reprezintă de fapt darea în factor comun a elementului pivot;
reducerea

$$\left.\begin{array}{l} a_{ij} = a_{ij} - a_{ik} \cdot a_{kj} \\ \text{pentru } j = n + m,\ \ldots,\ k \end{array}\right\} \begin{array}{l} \text{pentru} \\ i = 1,\ 2,\ \ldots,\ n;\ i \neq k. \end{array}$$

Organigrama subprogramului (exceptînd construcția matricei A) este dată în fig. 2.8.

Programul corespunzător este compus din următoarele instrucțiuni:

```
      SUBRØUTINE GAUJØR(N,M,B,U,X)
      DIMENSIØN B(20,20), U(20,20), X(20,20), A(20,40)
      DETER = 0.0
      NPM = N + M
      NP1 = N + 1
```

Acum se construiește matricea A:

```
      DØ 1 J = 1,N
      DØ 1K = 1,N
    1 A(J,K) = B(J,K)
      DØ 2 J = 1,N
      DØ 2 K = NP1,NPM
      KMN = K - N
    2 A(J,K) = U(J,KMN)
```

Din acest moment se poate începe eliminarea propriu-zisă:

```
      DØ 5 K = 1,N
      DETER = DETER*A(K,K)
      IF (ABS(A(K,K)).LT.0.0001)GØ TØ 7
```


Scanned with OKEN Scanner

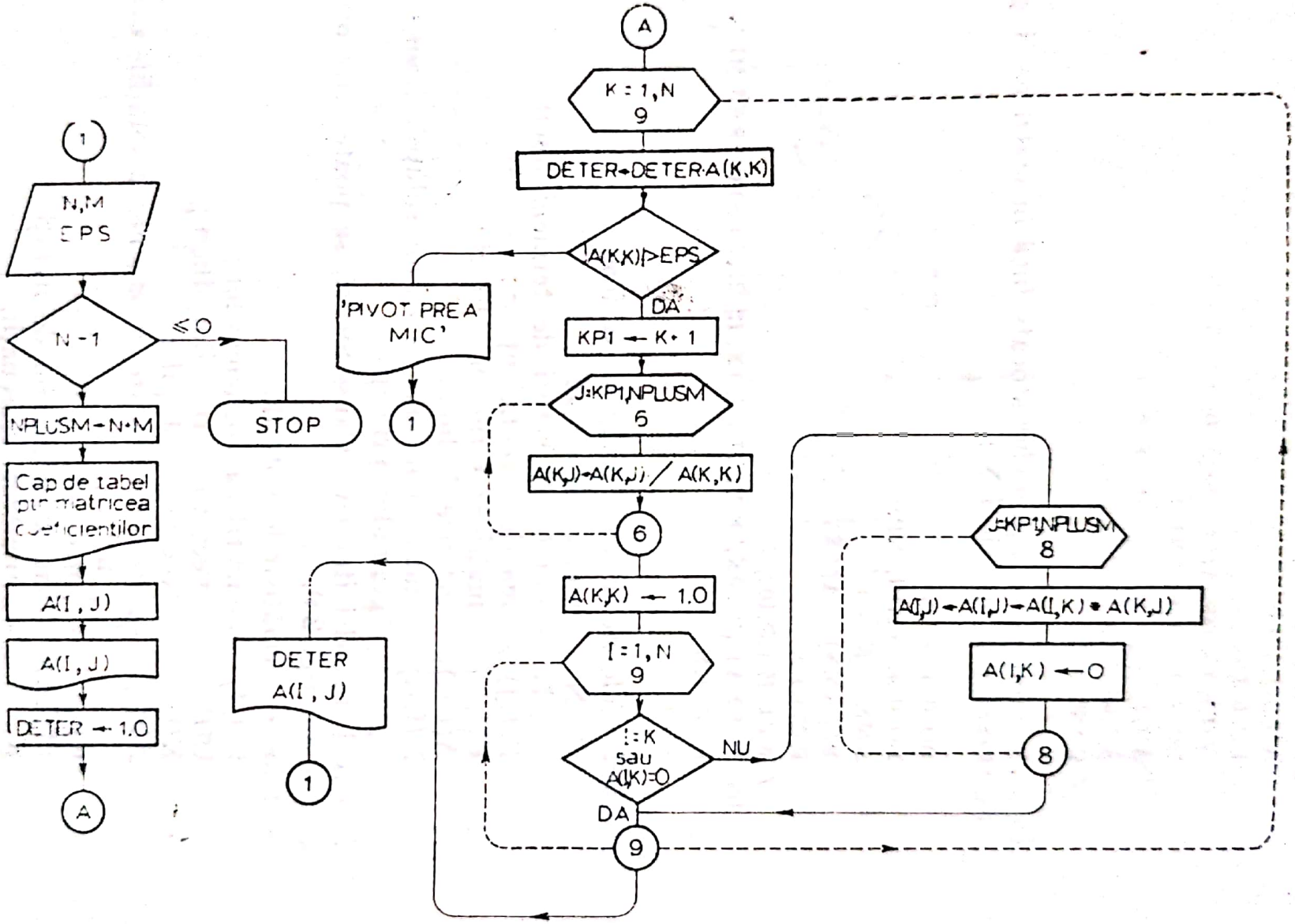


Fig. 2.8

Scanned with OKEN Scanner

```
      KP1 = K + 1
      DØ 3 J = KP1, NPM
    3 A(K,J) = A(K,J)/A(K,K)
      A(K,K) = 1.0
      DØ 5 I = 1,N
      IF(I.EQ.K.ØR.A(I,K).EQ.0.0) GØ TØ 5
      DØ 4 J = KP1, NPM
    4 A(I,J) = A(I,J) - A(I,K)*A(K,J)
      A(I,K) = 0.0
    5 CØNTINUE
```

În acest moment soluţiile sînt stocate încă în matricea A şi trebuie transferate matricei X:

```
      DØ 6 J = 1,N
      DØ 6 K = NP1, NPM
      KMN = K - 1
    6 X(J,KMN) = A(J,K)
      RETURN
    7 WRITE (IØ,100)K
  100 FØRMAT (//, 20X,' PIVØTUL ', 1X, I2, 2X,' ESTE PREA MIC')
      RETURN
      END
```

Notaţiile folosite în acest program au fost:

N — număr de ecuaţii;
M — număr de seturi de termeni liberi;
B(I,J) — matricea coeficienţilor;
U(I,K) — matricea termenilor liberi;
A(I,K) — matrice de lucru;
X(I,K) — matricea în care se trimit soluţiile înapoi programului principal.

În cazul metodei iterative Gauss-Siedel, se poate scrie organigrama din fig. 2.9.

Se folosesc următoarele notaţii:

B(I,J) — matricea coeficienţilor;
U(I) — vectorul termenilor liberi;
X(I) — vectorul soluţiilor;
XP(I) — vectorul în care se stochează soluţiile nedefinitive;
EPS — precizie;
ITMAX — numărul maxim de iteraţii;
N — numărul de ecuaţii.

Scanned with OKEN Scanner

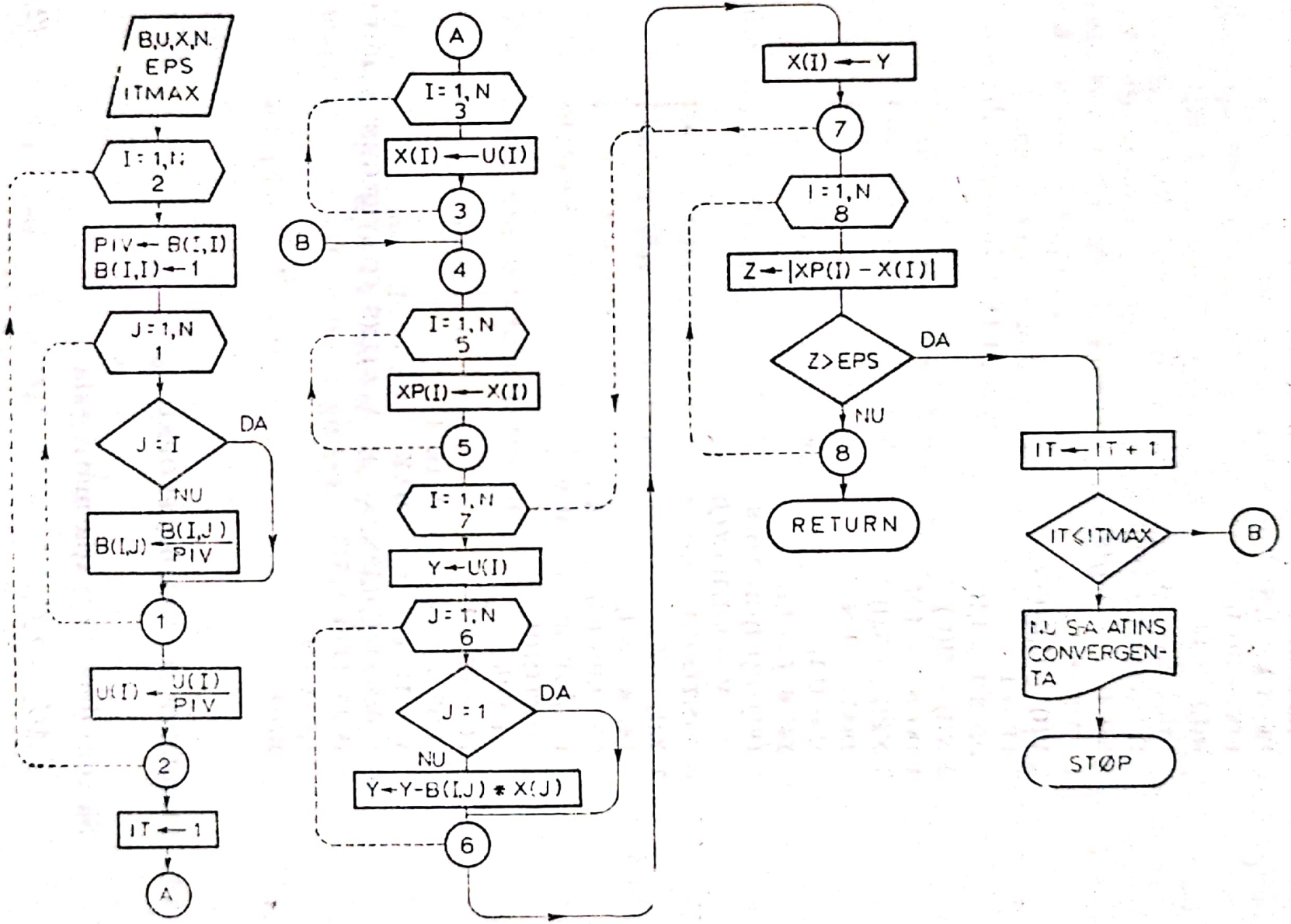


Fig. 2.9

Scanned with OKEN Scanner

Lista de instrucţiuni a programului este:

```
      SUBRØUTINE GASID (B,U,X,N,EPS, ITMAX)
      DIMENSIØN B(N,N), U(N), X(N), XP(N)
      DØ 2 I = 1,N
      PIV = B(I,I)
      B(I,I) = 1.0
      DØ 1 J = 1,N
      IF(J.EQ.I)GØ TØ 1
      B(I,J) = B (I,J)/PIV
    1 CØNTINUE
    2 U(I) = U(I)/PIV
      IT = 1
      DØ 3 I = 1,N
    3 X(I) = U(I )
    4 DØ 5 I = 1,N
    5 XP(I) = X(I)
      DØ 7I = 1,N
      Y = U(I)
      DØ 6 J = 1,N
      IF(J.RQ.I)GØ TØ 6
      Y = Y − B(I,J) *X(J)
    6 CØNTINUE
    7 X(I) = Y
      DØ 8 I = 1,N
      Z = ABS (XP(I) − X(I))
      IF(Z.GT.EPS)GØ TØ 9
    8 CØNTINUE
      RETURN
    9 IT = IT + 1
      IF(IT·LE·ITMAX)GØ TØ 4
      WRITE (IØ,100) ITMAX
  100 FØRMAT (////,10X,'NU S−A ATINS CØNVERGENTA DUPA',
     1 I5,2X,'ITERATII')
      RETURN
      END
```

2.4.5. PROBLEME DE VALORI PROPRII

Să considerăm ecuaţia matriceală

$$AC_i = \lambda_i C_i, \qquad (2.4.12)$$

Scanned with OKEN Scanner

unde:

λ_i este o constantă numerică (scalar)

C_i — un vector coloană.

λ_i şi C_i se numesc valoare proprie şi respectiv vector propriu aparţinînd matricei A.

Dacă vectorii C aparţin unui spaţiu liniar vectorial, atunci ecuaţia (2.4.12) (numită şi *problemă de valori proprii*) are ca semnificaţie separarea unui anumit set de vectori din acest spaţiu care atunci cînd sînt înmulţiţi cu matricea A se obţine un vector coliniar cu cel original.

Dacă A este o matrice $n \times n$ atunci ecuaţia (2.4.12) corespunde sistemului de ecuaţii:

$$\begin{cases} a_{11}c_1 + a_{12}c_2 + \ldots + a_{1n}c_n = \lambda_i c_1 \\ a_{21}c_1 + a_{22}c_2 + \ldots + a_{2n}c_n = \lambda_i c_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}c_1 + a_{n2}c_2 + \ldots + a_{nn}c_n = \lambda_i c_n \end{cases}$$

Rearanjînd, obţinem:

$$\begin{cases} (a_{11} - \lambda_i)c_1 + a_{12}c_2 + \ldots + a_{1n}c_n = 0 \\ a_{21}c_1 + (a_{22} - \lambda_i)c_2 + \ldots + a_{2n}c_n = 0 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}c_1 + \ldots\ldots\ldots + (a_{nn} - \lambda_i)c_n = 0 \end{cases} \qquad (2.4.13)$$

Trecînd din nou în formă matriceală

$$(A - \lambda_i 1)\, C_i = 0.$$

O soluţie a sistemului este $C_i = 0$, însă aceasta este o soluţie trivială. Condiţia ca sistemul să aibă soluţii netriviale este ca matricea $A - \lambda_i 1$ să fie singulară, adică

$$\det (A - \lambda_i 1) = 0,$$

adică

$$\begin{vmatrix} a_{11} - \lambda_i & a_{22} \ldots a_{1n} \\ a_{21} & a_{22} - \lambda_i \ldots a_{2n} \\ \ldots\ldots\ldots\ldots \\ a_{n1} & a_{n2} & a_{nn} - \lambda_i \end{vmatrix} = 0$$

Scanned with OKEN Scanner

Acest determinant se numeşte *determinant secular*. Dezvoltînd, se obţine o ecuaţie de gradul n în λ_i:

$$\lambda_i^n + \alpha_1 \lambda_i^{n-1} + \ldots + \alpha_{n-1}\lambda_i + \alpha_n = 0. \qquad (2.4.14)$$

Această ecuaţie se numeşte *ecuaţie caracteristică*. Rezolvînd-o se obţin n valori pentru λ_i, nu în mod necesar distincte, numite *valori proprii* ale matricei A.

Pentru fiecare valoare a lui λ_i se rezolvă apoi sistemul (2.4.13), obţinîndu-se astfel *vectorii proprii* corespunzători:

$$C_i = \begin{pmatrix} c_{1i} \\ c_{2i} \\ \ldots \\ c_{ni} \end{pmatrix}.$$

Dacă toate valorile C_i sînt distincte (nu se repetă nici o rădăcină a ecuaţiei (2.4.14)), atunci se spune că matricea A este *nedegenerată*.
Notînd:

$$C = [C_1 C_2 \ldots C_n] = \begin{pmatrix} c_{11} & c_{21} & \ldots & c_{1n} \\ c_{21} & c_{22} & \ldots & c_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ c_{n1} & c_{n2} & \ldots & c_{nn} \end{pmatrix}; \Lambda = \begin{pmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & \lambda_n \end{pmatrix}$$

atunci toate sistemele de tipul 2.4.13 se pot reuni în ecuaţia matricială

$$AC = \Lambda C,$$

sau dacă $\det(C) \neq 0$,

$$C^{-1} AC = \Lambda,$$

adică spunem că matricea A este diagonalizată de matricea vectorilor săi proprii.

Găsirea vectorilor proprii care diagonalizează o matrice este de fapt etapa esenţială în probleme legate de determinarea structurii electronice a moleculelor. Una din metodele simple de rezolvare a problemei de valori proprii este chiar cea adoptată în prezentarea ei. Această metodă este însă neeconomică, ţinînd cont că de cele mai multe ori matricea A are dimensiuni peste 10×10, ceea ce ar însemna rezolvarea a mai mult de 10

Scanned with OKEN Scanner

sisteme de ecuații liniare fără a mai pune la socoteală obținerea și rezolvarea ecuației caracteristice.

Există metode numerice care rezolvă această problemă direct prin aproximări succesive ale setului de vectori proprii. Două astfel de metode sînt metoda Jacobi și metoda Givens. Modurile de lucru, algoritmii acestor programe sînt deosebit de complicate și nu merită a fi expuse. Ceea ce trebuie cunoscut este modul de folosire al acestor programe.

Subprogramul JACØBI are următoarea secvență de chemare:

```
CALL JACØBI(H,N,U)
```

unde:

H este matricea de diagonalizat;
N — dimensiunea matricii de diagonalizat;
U — matricea vectorilor proprii;

La sfîrșitul calculelor, valorile proprii vor fi stocate în diagonala matricei inițiale H.

Subprogramul GIVENS are următoarea secvență de chemare:

```
CALL GIVENS(NX,NRØØTX,NJXA,B,RØØT, VECT)
```

unde

NX este dimensiunea matricei de diagonalizat;
NRØØTX — numărul de valori proprii dorite;
NJX — dimensiunea vectorilor proprii;
A — matrice în formă triunghiulară, stocată pe coloane;
RØØT — valorile proprii;
VECT — vectorii proprii.

Matricea *A* se obține din matricea de diagonalizat cu următoarele instrucțiuni:

```
      LL = 1
      DØ 8 J = 1,NX
      DØ 8 I = 1,J
      A(LL) = H(J,I)
    8 LL = LL + 1
```

Se dau în continuare listele de instrucțiuni ale celor două subprograme:

```
      SUBRØUTINE JACØBI (/H/, /N/, /U/)
      DIMENSIØN H(21,21), U(21, 21), X(21), IQ(21)
```


Scanned with OKEN Scanner

```
   10 IF(1.GT.N) GØTØ79044
      DØ141 = 1, N
      IF(1.GT.N) GØTØ79044
      DØ14J = 1, N
      IF(I - J) 12,11,12
   11 U(I,J) = 1.
      GØTØ14
   12 U(I,J) = 0.0
   14 CØNTINUE
79044 CØNTINUE
   15 NR = 0
      IF(N-1) 1000,1000,17
   17 NMII = N - 1
      IF(1.GT.NMI 1) GØTØ79045
      DØ30I = 1, NMII
      X(I) = 0.0
      IPL1 = I + 1
      IF(IPL1.GT.N) GØTØ79045
      DØ30J = IPL1, N
      IF(X(I) - ABS(H(I,J)))20,20,30
   20 X(I) = ABS(H(I,J))
      IQ(I) = J
   30 CØNTINUE
79045 CØNTINUE
      RAP = 1./(2.**36)
      HDTEST = 1.E38
   40 IF(1.GT.NMI1) GØTØ79046
      DØ70I = 1, NMI1
      IF(I-1) (60,60,45
   45 IF(XMAX-X(I) 60,70,70
   60 XMAX = X(I)
      IPIV = I
      JPIV = IQ(I)
   70 CØNTINUE
79046 CØNTINUE
      IF(XMAX) 1000,1000,80
   80 IF(HDTEST)90,90,85
   85 IF(XMAX-HDTEST)90,90,148
   90 HDIMIN = ABS(H(1,1))
      IF(2.GT.N) GØTØ79047
      DØ110I = 2,N
      IF(HDIMIN-ABS(H(I,I)))110,110,100
  100 HDIMIN = ABS(H(I,I))
```


Scanned with OKEN Scanner

```
  110 CØNTINUE
79047 CØNTINUE
      HDTEST = HDIMIN*RAP
      IF(HDTEST-XMAX) 148,1000,1000
  148 NR = NR + 1
  150 TANG = SIGN(2.0000, (H(IPIV, IPIV) - H(JPIV, JPIV)))
    1 H(IPIV, JPIV)/ABS(H(IPIV,IPIV)-H(JPIV, JPIV)) + SQRT ((H
    * (IPIV, IPIV)-H(JPIV, JPIV))** 2 + 4·*H(IPIV, JPIV)**2))
      CØSINE = 1./SQRT (1. + TANG**2)
      SINE = TANG*CØSINE
      HII = H(IPIV, IPIV)
      H(IPIV, IPIV) = CØSINE**2* (HII + TANG*(2.*H(IPIV,
    * JPIV) + TANG*H (JPIV, JPIV)))
      H(JPIV,JPIV) = CØSINE**2*(H(JPIV, JPIV) - TANG*
    * (2.*H(IPIV, JPIV) - TANG*HII))
      H(IPIV, JPIV) = 0.0
      IF(H(IPIV, IPIV)-H(JPIV, JPIV)) 152, 153, 153
  152 HTEMP = H(IPIV, IPIV)
      H(IPIV, IPIV) = H(JPIV,JPIV)
      H(JPIV, JPIV) = HTEMP
      HTEMP = SINGN (1.0000, - SINE)*CØSINE
      CØSINE = ABS(SINE)
      SINE = HTEMP
  153 CØNTINUE
      IF(1.GT.NMI1) GØTØ79048
      DØ350I = 1, NMI1
      IF(I - IPIV) 210,350, 200
  200 IF(I-JPIV) 210,350, 210
  210 IF(IQ(I) - IPIV) 230, 240, 230
  230 IF(IQ(I) - JPIV)350,240, 350
  240 K = IQ(I)
  250 HTEMP = H(I,K)
      H(I,K) = 0.0
      IPL1 = I + 1
      X(I) = 0.0
      IF(IPL1.GT.N) GØTØ79049
      DØ320J = IPL1,N
      IF(X(I) - ABS(H(I,J))) 300,300, 320
  300 X(I) = ABS(H(I,J))
      IQ(I) = J
  320 CØNTINUE
79049 CØNTINUE
      H(I,K) = HTEMP
```


Scanned with OKEN Scanner

```
  350 CØNTİNUE
79048 CØNTINUE
      X(IPIV) = 0.0
      X(JPIV) = 0.0
      IF(1.GT.N) GØTØ79050
      DØ5301 = 1, N
      IF(I – IPIV) 370,530, 420
  370 HTEMP = H(I,IPIV)
      H(I,IPIV) = CØSINE*HTEMP + SINE*H(I, JPIV)
      IF(X(I) – ABS(H(I,IPIV)))380,390,390
  380 X(I) = ABS(H(I,IPIV))
      IQ(I) = IPIV
  390 H(I, JPIV) = –SINE*HTEMP + CØSINE*H (I, JPIV)
      IF(X(I)–ABS(H(I, JPIV))) 400,530, 530
  400 X(I) = ABS(H(I, JPIV))
      IQ(I) = JPIV
      GØTØ530
  420 IF(I– JPIV)430,530,480
  430 HTEMP = H(IPIV,I)
      H(IPIV, I) =CØSINE*HTEMP + SINE*H(I, JPIV)
      IF(X(IPIV)–ABS (H(IPIV, I))) 440, 450, 450
  440 X(IPIV) = ABS(H(IPIV,I))
      IQ(IPIV) = I
  450 H(I, JPIV) = –SINE*HTEMP + CØSINE*H (I, JPIV)
      IF(X(I)–ABS(H(I, JPIV)))400,530, 530
  480 HTEMP = H(IPIV, I)
      H(IPIV, I) = CØSINE*HTEMP + SINE*H(JPIV, I)
      IF(X(IPIV)–ABS(H(IPIV, I))) 490, 500, 500
  490 X(IPIV) = ABS (H(IPIV,I))
      IQ(IPIV) = I
  500 H(JPIV,I) = – SINE*HTEMP + CØSINE*H(JPIV,I)
      IF(X(JPIV) – ABS(H(JPIV,I))) 510,530,530
  510 X(JPIV) = ABS (H(JPIV, I))
      IQ(JPIV) = I
  530 CØNTINUE
79050 CØNTINUE
  540 IF(1.GT.N) GØTØ79051
      DØ550I = 1,N
      HTEMP = U(I,IPIV)
      U(I,IPIV) = CØSINE*HTEMP + SINE*U(I, JPIV)
  550 U(I, JPIV) = – SINE*HTEMP + CØSINE*U(I, JPIV)
79051 CØNTINUE
      GØTØ40
```


Scanned with OKEN Scanner

```
1000 CØNTINUE
     NM = N - 1
     IF(1.GT.NM) GØTØ79052
     DØ740J = 1,NM
     JP = J + 1
     IF(1.GT.J) GØTØ79053
     DØ743K = 1,J
     IF(H(K,K) - H(JP,JP))741, 743, 743
 741 CØNTINUE
     HH = H(K,K)
     H(K,K) = H(JP,JP)
     H(JP,JP) = HH
     IF(1.GT.N) GØTØ79054
     DØ742I = 1,N
     X(I) = U(I,K)
     U(I,K) = U(I,JP)
     U(I,JP) = X(I)
 742 CØNTINUE
79054 CØNTINUE
 743 CØNTINUE
79053 CØNTINUE
 740 CØNTINUE
79052 CØNTINUE
     RETURN
     END
     SUBRØUTINE GIVENS(NX,NRØØTX,NJX,A,B,RØØT,VECT)
     DIMENSIØN B(NX,5),A(1),RØØTX(NRØØTX), VECT(NJX, NRØØTX)
     ETA = 1.E-16
     THETA = 1.E 75
     DEL1 = ETA/100.
     DELTE = ETA**2*100.
     SMALL = ETA**2/100.
     DELBIG = THETA*DELTA/1000.
     THETA1 = 1000./THETA
     TØLER = ETA*100.
     RPØWER = 8388608.
     RPØW1 = RPØWER/2.
     RAND1 = PRØWER-2.
     N = NX
     NRØØT = IABS(NRØØTX)
     IF(NRØØT.EQ.O)GØ TØ 1001
     IF(N-1)1001,1003,105
1003 RØØT(1) = A(1)
```


Scanned with OKEN Scanner

```
      IF(NRØØTX.GT.O) VECT(1,1) = 1.0
      GØ TØ 1001
  105 CØNTINUE
      NSIZE = (N*(N+1))/2
      NM1 = N-1
      NM2 = N-2
      FACTØR = 0.
      DØ 70 I = 1,NSIZE
   70 FACTØR = AMAX1(FACTØR,ARS(A(I)))
      IF(FACTØR.NE.O)GØ TØ 72
      DØ 78 I = 1,NRØØT
      IF(NRØØTX.LT.0)GØ TØ 78
      DØ 77 J = 1,N
   77 VECT(J,I) = 0.
      VECT(I,I) = 1.0
   78 RØØT(I) = 0.
      GØ TØ 1001
   72 ANØRM = 0.
      J = 1
      K = 1
   86 DØ 80 I = 1,NSIZE
      IF(I.NE.J)GØ TØ 81
      ANØRM = ANØRM+(A(I)/FACTØR)**2/2.
      K = K+1
      J = J+K
      GØ TØ 80
   81 ANØRM = ANØRM+(A(I)/FACTØR)**2
   80 CØNTINUE
   83 ANØRM = SQRT(ANØRM*2.)*FACTØR
      DØ 91 I = 1,NSIZE
   91 A(I) = A(I)/ANØRM
      ALIMIT = 1.0
      ID = 0
      IA = 1
      IF(NM2.EQ.0)GØ TØ 201
      DØ 200 J = 1,NM2
      IA = IA+J+2
      ID = ID+J+1
      JP2 = J+2
      II = IA
      SUM = 0.0
```


Scanned with OKEN Scanner

```
      DØ 100 I = JP2,N
      SUM = SUM+A(II)**2
  100 II = II+I
      TEMP = A(ID)
      IF(SUM.GT.SMALL)GØ TØ 110
  120 B(J.1) = TEMP
     A(ID) = 0.
      GØ TØ 200
  110 SUM = SQRT(SUM+TEMP**2)
      B(J,1) = -SIGN(SUM,TEMP)
      B(J+1,2) = SQRT((1.0+ABS(TEMP)/SUM)/2.0)
      TEMP = SIGN(0,5/(B(J+1,2)*SUM),TEMP)
      II = IA
      DØ 130 I = JP2,N
      B(I,2) = A(II)*TEMP
  130 II = II+I
      AK = 0.0
      IC = ID+1
      J1 = J+1
      DØ 190 I = J1,N
      JJ = IC
      TEMP = 0.
      DØ 180 II = J1,N
      TEMP = TEMP + B(II,2)*A(JJ)
      IF(II.LT.I)  GØ TØ 210
  140 JJ = JJ + II
      GØ  TØ 180
  210 J J = JJ + 1
  180 CØNTINUE
      AK = AK + TEMP*B(I,2)
      B(I,1) = TEMP
  190 IC = IC + I
      DØ  150 I = J1,N
  150 B(I,1) = B(I,1) - AK*B(I,2)
      JJ = ID
      DØ  160 I = J1,  N
      A(JJ) = B(I,2)
      DØ  170 II = J1,I
      JJ = JJ + 1
  170 A(JJ) = A(JJ)-2·0*(B(I,1)*B(II,2) + B(I,2)*B(II,1))
  160 JJ = JJ + J
```


Scanned with OKEN Scanner

```
200 CØNTINUE
201 CONTINUE
    B(NM1,1) = A(NSIZE-1)
    A(NSIZE-1) = 0.
    JUMP = 1
    DØ 320 J = 1,N
    B(J,2) = A(JUMP)
    B(J,3) = B(J,1)**2
320 JUMP = JUMP + J + 1
    DØ 310 I = 1,NRØØT
310 RØØT(I) = ALIMIT
    RØØTL = -ALIMIT
    DØ 330 I = 1, NRØØT
    RØØTX = + ALIMIT
    DØ 335 J = I, NRØØT
335 RØØTX = AMIN1 (RØØTX, RØØT(J))
    RØØT(I) = RØØTX
500 TRIAL = (RØØTL + RØØT(I))*0.5
    IF(TRIAL.EQ.RØØTL.ØR.TRIAL.EQ.RØØT(I)) GØ TØ 330
350 CØNTINUE
    NØMTCH = N
    J = 1
360 FØ = B(J,2) - TRIAL
370 CØNTINUE
    IF(ABS(FØ).LT.THETA1)GØ TØ 380
     IF(FO.GE.0.) NØMTCH = NØMTCH-1
    J = J + 1
    IF(J.GT.N) GØ TØ 390
    FO = B(J,2) - TRIAL-B(J-1,3)/FØ
    GØ TØ 370
380 J = J + 2
    NØMTCH = NØMTCH - 1
    IF(J.LE.N) GØ TØ 360
 90 CØNTINUE
    IF(NØMTCH.GE.I) GØ TØ 540
    RØØTL = TRIAL
    GØ TØ 500
540 RØØT(I) = TRIAL
    NØM = MIN0 (NRØØT,NØMTCH)
    RØØT(NØM) = TRIAL
    GØ TØ 500
330 CØNTINUE
    NRT = NRØØT/2
```


Scanned with OKEN Scanner

```
      DØ 10 I = 1, NRT
      SAVE = RØØT(I)
      NMIP1 = NRØØT - I + 1
      RØØT(I) = RØØT(NMIP1)
   10 RØØT (NMIP1) = SAVE
  807 CØNTINUE
      IF (NRØØTX·LT·0) GØ TØ 1002
      DØ 15 I = 1,N
      DØ 15 J = 1, NRØØT
   15 VECT (I,J) = 1.0
      DØ 700 I = 1, NRØØT
      ARØØT = RØØT (I)
      IF(I.EQ.1) GØ TØ 710
  715 IF(ABS(RØØT (I - 1) - ARØØT).LT.TØLER) GØ TØ 720
  710 IA = - 1
  720 IA = IA + 1
      ELIM1 = A(1) - ARØØT
      ELIM2 = B(1,1)
      JUMP = 1
      DØ 750 J = 1, NM1
      JUMP = JUMP + J + 1
      IF(ABS(ELIM1). LE.ABS(B(J,1))) GØ TØ 760
      B(J,2) = ELIM1
      B(J,3) = ELIM2
      B(J,4) = 0.0
      TEMP = B(J,1)/ELIM1
      ELIM1 = A (JUMP) - ARØØT - TEMP*ELIM2
      ELIM2 = B (J + 1,1)
      GØ TØ 755
  760 B(J,2) = B(J,1)
      B(J,3) = A(JUMP) - ARØØT
      B(J,4) = B(J + 1,1)
      TEMP = 1·0
      IF(ABS(B(J,1)).GT.THETA1) TEMP = ELIM1/B(J,1)
      ELIM1 = ELIM2-TEMP*B(J,3)
      ELIM2 = - TEMP*B(J + 1,1)
  755 B(J,5) = TEMP
  750 CØNTINUE
      B(N,2) = ELIM1
      B(N,3) = 0.
      B(J,4) = 0.
      B(NM1,4) = 0
```


Scanned with OKEN Scanner

```
      ITER = 1
      IF(IA.NE.0) GØ TØ 801
  790 L = N + 1
      DØ 780 J = 1,N
      L = L - 1
  786 CØNTINUE
      ELIM1 = VECT (L,I) - VECT (L + 1, I)*B(L,3) - VECT
      1 (L + 2, I)*B(L,4)
      IF (ABS(ELIM1).GT.DELBIG) GØ TØ 782
      TEMP = B(L,2)
      IF(ABS (B(L,2)).LT.DELTA) TEMP = DELTA
      VECT (L,I) = ELIM1/TEMP
      GØ TØ 780
  782 DØ 784 K = 1,N
  784 VECT(K,I) = VECT(K,I)/DELBIG
      GØ TØ 786
  780 CØNTINUE
      GØ TØ (820, 800), ITER
  820 ITER = ITER + 1
  890 ELIM1 = VECT(1,I)
      DØ 830 J = 1, NM1
      IF(B(J,2).EQ.B(J,1)) GØ TØ 840
  850 VECT(J,I) = ELIM1
      ELIM1 = VECT(J + 1,I) - ELIM1*B(J,5)
      GØ TØ 830
  840 VECT(J,I) = VECT (J + 1,I)
      ELIM1 = ELIM1-VECT(J + 1,I)*TEMP
  830 CØNTINUE
      VECT(N,I) = ELIM1
      GØ TØ 790
  801 CØNTINUE
      DØ 802 J = 1,N
      RAND1 = AMØD(4099**RAND1, RPØWER)
  802 VECT(J,I) = RAND1/RPØW1 - 1.
      GØ TØ 790
  800 IF(IA.EQ.0) GØ TØ 885
      DØ 860 J1 = 1,IA
      K = I-J1
      TEMP = 0.
      DØ 870 J = 1,N
  870 TEMP = TEMP + VECT(J,I)*VECT(J,K)
      DØ 880 J = 1,N
```


Scanned with OKEN Scanner

```
880 VECT(J,I) = VECT(J,I) - TEMP*VECT(J,K)
860 CONTINUE
885 GO TO (890,900), ITER
900 ELIM1 = 0.
    DO 904 J = 1,N
904 ELIM1 = AMAX1 (ABS(VECT(J,I)), ELIM1)
    TEMP = 0.
    DO 910 J = 1,N
    ELIM2 = VECT(J,I)/ELIM1
    TEMP = TEMP + ELIM2**2
910 CONTINUE
    TEMP = 1.0/(SQRT (TEMP)*ELIM1)
    DO 920 J = 1,N
    VECT(J,I) = VECT(J,I)*TEMP
    IF(ABS(VECT(J,I)).LT.DEL1) VECT(J,I) = 0.
920 CONTINUE
700 CONTINUE
    IF(NM2.EQ.0) GO TO 1002
    JUMP = NSIZE-(N + 1)
    IM = NM1
    DO 950 I = 1,NM2
    J1 = JUMP
    DO 955 J = IM,N
    B(J,2) = A(J1)
955 J1 = J1 + J
    DO 960 K = 1, NROOT
    TEMP = 0.
    DO 970 J = IM,N
970 TEMP = TEMP + B(J,2) *VECT(J,K)
    TEMP = TEMP + TEMP
    DO 980 J = IM,N
980 VECT(J,K) = VECT(J,K) - TEMP*B(J,2)
960 CONTINUE
    JUMP = JUMP-IM
950 IM = IM - 1
1002 CONTINUE
    DO 95 I = 1, NROOT
 95 ROOT(I) = ROOT(I)*ANORM
1001 RETURN
    END
```


Scanned with OKEN Scanner

2.5 REZOLVAREA NUMERICĂ A ECUAȚIILOR DIFERENȚIALE CU DERIVATE PARȚIALE

Cele mai frecvent întîlnite ecuații diferențiale cu derivate parțiale au forma (sau pot fi aduse la forma) :

$$\sum_{i=1}^{n} A_i \frac{\partial^2 u}{\partial x_i^2} + \sum_{i=1}^{n} B_i \frac{\partial u}{\partial x_i} + Cu + D = 0.$$

Următoarele posibilități sînt cele mai importante :

— toți $A_i \neq 0$ și de același semn — ecuație eliptică ;

— toți $A_i \neq 0$ dar unul de semn diferit de toți ceilalți și atunci ecuația se numește ecuație hiperbolică ;

— toți A_i, exceptînd unul ($A_k = 0$), diferiți de zero iar $B_k \neq 0$ și de semn contrar cu A_i — ecuație parabolică.

În majoritatea cazurilor ce se întîlnesc în practică una dintre variabilele independente este timpul iar celelalte sînt coordonatele spațiale.

Pentru rezolvarea numerică a acestor ecuații se utilizează transformarea lor în ecuații cu diferențe finite. Pentru aceasta, după stabilirea condițiilor limită, se construiește o rețea de noduri peste regiunea de variație a variabilelor independente. Pentru diversele variabile se aleg pașii Δx_j și atunci orice nod din rețea va avea coordonate de forma $j_\alpha \Delta x_\alpha$.

Vom nota cu $u_{j_1, j_2, \ldots, j_n}$ soluția exactă în punctul de coordonate $(j_1^{x_1}, \ldots, j_n^{x_n})$ iar cu $v_{j_1, \ldots, j_n}$ soluția aproximativă obținută aplicînd metoda diferențelor finite. Dacă rețeaua de noduri este suficient de deasă putem spera ca valorile $v_{j_1, \ldots, j_n}$ să tindă spre $u_{j_1, \ldots, j_n}$.

Considerăm o ecuație cu numai două variabile independente. În acest caz se folosesc următoarele transformări în diferențe finite

$$\frac{\partial u}{\partial x} = \frac{u_{i+1,j} - u_{i,j}}{\Delta x} \quad \text{— diferență înainte}$$

$$\frac{\partial u}{\partial x} = \frac{u_{i,j} - u_{i-1,j}}{\Delta x} \quad \text{— diferență în urmă}$$

$$\frac{\partial u}{\partial x} = \frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x} \quad \text{— diferență centrală}$$

$$\frac{\partial u}{\partial x^2} = \frac{u_{i-1,j} - 2u_{i,j} - u_{i+1,j}}{(\Delta x)^2}.$$

Scanned with OKEN Scanner

Formule similare se pot scrie şi pentru a doua variabilă independentă (y). De asemenea,

$$\frac{\partial^2 u}{\partial x \partial y} = \frac{u_{i+1,j+1} - u_{i-1,j+1} - u_{i+1,j-1} + u_{i-1,j-1}}{4\Delta x \Delta y}.$$

2.5.1 UTILIZAREA DIFERENŢELOR FINITE

Cel mai simplu mod de folosire a ecuaţiilor cu diferenţe finite constă în calculul pas cu pas al valorilor v pornind de la valorile iniţiale şi/sau limită cunoscute. Această metodă se numeşte metoda explicită.

De exemplu, pentru ecuaţia:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x}, \tag{2.5.1}$$

cu condiţiile limită:

$$u(x,0) = f(x) \text{ pt.} 0 \leqslant x \leqslant 1$$
$$u(0,t) = g_0(t) \text{ pt. } 0 \leqslant t \leqslant T$$
$$u(1,t) = g_1(t) \text{ pt. } 0 \leqslant t \leqslant T$$

putem scrie următoarele ecuaţii cu diferenţe finite

$$v_{i,n+1} = \lambda v_{i-1,n} + (1-2\lambda)v_{i,n} + \lambda v_{i+1,n}$$
$$v_{0,n+1} = g_0(t_{n+1})$$
$$v_{N,n+1} = g_1(t_{n+1})$$
$$v_{i,0} = f(X_i)$$

Aceste ecuaţii descriu distribuţia temperaturii în timp într-o bară ale cărei capete se află fixate în instalaţii care permit reglarea temperaturii după o funcţie oarecare de timp.

În ecuaţiile de mai sus, $\lambda = \Delta t/(\Delta x)^2$. Pentru a minimiza eroarea este necesar ca λ să fie cuprins între 0 şi $^1/_2$. Aceasta face ca timpul de calcul să crească foarte mult. De asemenea se observă din fig. 2.10 că metoda aceasta neglijează efectul întregii regiuni A asupra valorii $v_{i,n}$.

Cele două inconveniente enumerate mai sus sînt înlăturate în cazul metodei implicite, cînd derivata de ordinul doi (referindu-ne tot la ecuaţia 2.5.1) se calculează la un nod avansat

Scanned with OKEN Scanner

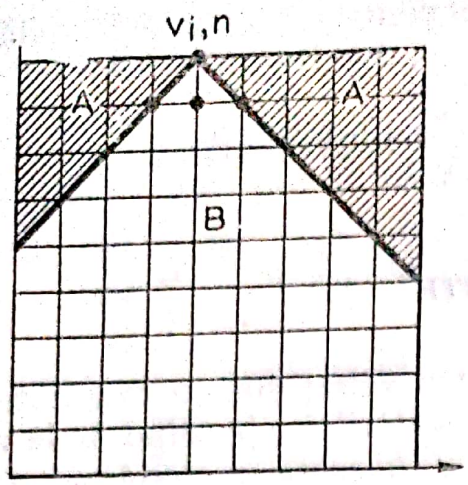


Fig. 2.10

în timp cu o treaptă faţă de nodul în care se calcula în cazul metodei explicite:

$$\frac{v_{i,n+1} - v_{i,n}}{\Delta t} = \frac{v_{i-1,\,n+1} - 2v_{i,n+1} + v_{i+1,n+1}}{(\Delta x)^2}.$$

Prin rearanjare şi introducînd din nou pe λ obţinem:

$$-\lambda v_{i-1,n+1} + (1 + 2\lambda)v_{i,n+1} - \lambda v_{i+1,n+1} = v_{i,n}. \qquad (2.5.2)$$

Condiţiile limită rămîn aceleaşi ca pentru metoda explicită.

Dacă numărul de intervale Δx este $N - 1$ atunci vom putea scrie $N - 1$ ecuaţii de tipul 2.5.2:

$$(1 + 2\lambda)v_{1,n+1} - v_{2,n+1} = v_{1,n} + \lambda g_0\,(t_{n+1})$$

$$-\lambda v_{i-1,\,n+1} + (1+2\lambda)v_{i,\,n+1} - \lambda v_{i+1,\,n+1} = v_{i,\,n} \quad \text{pt.} 2 \leqslant i \leqslant N-2$$

$$-\lambda v_{N-2,\,n+1} + (1+2\lambda)v_{N-1,\,n+1} = V_{N-1,\,n} + \lambda g_1(t_{n+1}).$$

Acesta este de fapt un sistem liniar de forma:

$$\begin{cases} a_1v_1 + b_1v_2 = d_1 \\ c_2v_1 + a_2v_2 + b_2v_3 = d_2 \\ \qquad c_3v_2 + a_3v_3 + b_3v_4 = d_3 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \qquad c_{M-1}v_{M-2} + a_{M-1}v_{M-1} + b_{M-1}v_M = d_{M-1} \\ \qquad\qquad c_Mv_{M-1} + a_Mv_M = d_M \end{cases}$$

unde $M = N - 1$

Rezolvînd acest sistem printr-o metodă oarecare se obţin simultan toate valorile $v_{j,\,n+1}$.

Scanned with OKEN Scanner

Considerăm ca exemplu ecuaţiile care descriu funcţionarea unui reactor catalitic cu strat fix de catalizator. O formă simplificată a acestora ar fi :

$$\frac{\partial X_A}{\partial x} - A\left(\frac{\partial^2 X_A}{\partial r^2} + \frac{1}{r}\frac{\partial X_A}{\partial r}\right) - f_1(X_A, T) = 0$$

$$\frac{\partial T}{\partial X} - B\left(\frac{\partial^2 T}{\partial r^2} + \frac{1}{r}\frac{\partial T}{\partial r}\right) - f_2(X_A, T) = 0$$

Utilizînd metoda explicită şi respectiv implicită, ecuaţiile pentru X_A vor deveni (2.5.3) şi respectiv (2.5.4) :

$$\frac{X_{A_{i,j+1}} - X_{A_{i,j}}}{\Delta x} - A\left(\frac{X_{A_{i-1,j}} - 2X_{A_{i,j}} + X_{A_{i+1,j}}}{(\Delta r)^2} + \right.$$

$$\left. + \frac{1}{i\Delta r}\frac{X_{A_{i+1,j}} - X_{A_{i-1,j}}}{2\Delta r}\right) + f_1(X_{A_{i,j}}, T_{i,j}) = 0 \qquad (2.5.3)$$

$$\frac{X_{A_{i,j+1}} - X_{A_{i,j}}}{\Delta x} - A\left(\frac{X_{A_{i-1,j+1}} - 2X_{A_{i,j+1}} + X_{A_{i+1,j+1}}}{(\Delta r)^2} + \right.$$

$$\left. + \frac{1}{i\Delta r}\frac{X_{A_{i+1,j+1}} - X_{A_{i-1,j+1}}}{2\Delta r}\right) + f_1(X_{A_{i,j}}, T_{i,j}) = 0. \qquad (2.5.4)$$

Două ecuaţii similare vor fi scrise şi pentru T. Condiţiile limită în ambele cazuri sînt :

$$\begin{cases} X_{A_{0,j}} = 0 \\ T_{0,j} = T_0 \end{cases}$$

$$\begin{cases} r = 0 \quad \dfrac{\partial X_A}{\partial r} = 0 \quad \dfrac{\partial T}{\partial r} = 0 \\ r = 1 \quad \dfrac{\partial X_A}{\partial r} = 0 \quad \dfrac{\partial T}{\partial r} = -B_i(T - T_p). \end{cases}$$

Aceste condiţii vor modifica ecuaţiile de calcul pentru nodurile de reţea din regiunile limită. Vom nota X_A cu U^1 şi T cu U^2, A cu A_1, B cu A_2 şi atunci vom obţine următorul algoritm de calcul :

— pentru metoda explicită

$$U^1_{i,0} = 0 \quad U^2_{i,0} = T_0$$

$$U^k_{i,j+1} = U^k_{i,j}\left(1 - 2A_k\frac{\Delta x}{(\Delta r)^2}\right) + A_k\frac{\Delta x}{(\Delta r)^2 2i}((2i-1)U^k_{i-1,j} +$$

$$+ (2i+1)U^k_{i+1,j}) + f_k(U^1_{i,j}, U^2_{i,j})$$

pentru $1 < i < N - 1$, $j \neq 0$

— pentru $r = 0$, $i = 0$;

$$U^k_{0,j+1} = U^k_{0,j}\left(1 - 2A\frac{\Delta x}{(\Delta r)^2}\right) + 2A_k\frac{\Delta x}{(\Delta r)^2}U^k_{1,j} + f_k(U^k_{0,j})$$

— pentru $r = 1$, $i = N$;

$$U^1_{N,j+1} = U^1_{N,j}\left(1 - 2A\frac{\Delta x}{(\Delta r)^2}\right) + A_1\frac{\Delta x}{(\Delta r)^2}U^1_{N-1,j} + f_1(U^k_{N,j})$$

$$U^2_{N,j+1} = U^2_{N,j}.$$

Un algoritm similar se poate scrie şi pentru metoda implicită. Ca indicaţie, pentru stabilirea formulelor în condiţiile limită trebuie făcute substituţiile în ecuaţiile diferenţiale iniţiale. A nu se uita că se va obţine cîte un sistem de ecuaţii liniare pentru fiecare din cele două variabile dependente X_A şi T. În afară de metodele prezentate pînă aici mai există o serie de metode mai complicate care diferă de acestea şi diferă între ele prin alegerea nodurilor necesare pentru calculul unui nou nod.

— Metoda Crank-Nicolson (fig. 2.11)

Cu × am însemnat nodurile implicate în derivate în Z iar cu • am însemnat nodurile implicate în derivate în Y. Diferenţiala finită se construieşte astfel:

$$\frac{\partial U}{\partial Z} = \delta_z U_{i,n+1}$$

$$\frac{\partial^2 U}{\partial Y^2} = \frac{1}{2}\delta^2_Y U_{i,n+1} + \frac{1}{2}\delta^2_Y U_{i,n},$$

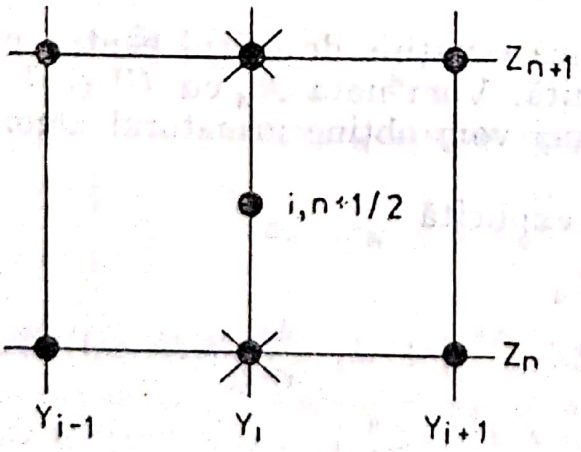


Fig. 2.11

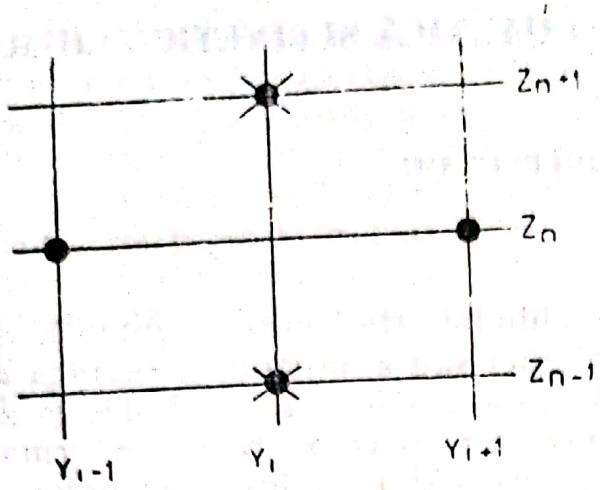


Fig. 2.12

unde operatorul δ_Y numit operator de diferență centrală are proprietățile :

$$\delta_Y^2 U_{i,j} = \frac{U_{i-1,j} - 2U_{i,j} + U_{i+1,j}}{(\Delta Y)^2}$$

$$\delta_Y U_{i,j} = \frac{U_{i+1/2,j} - U_{i-1/2,j}}{\Delta Y}.$$

Deci, în cazul acestei metode, ecuațiile se scriu față de punctul $i + {}^1/_2,\ j$.

— Metoda DuFort-Frankel (fig. 2.12)

$$\frac{dU}{dz} = \frac{U_{i,n+1} - U_{i,n-1}}{2\Delta z}$$

$$\frac{d^2U}{dy^2} = \frac{U_{i-1,n} - U_{i,n-1} - U_{i,n+1} + U_{i+1,n}}{(\Delta y)^2}.$$

Pentru această metodă este necesară cunoașterea a două nivele pentru calcularea celui de al treilea.

Scanned with OKEN Scanner

CAPITOLUL 3 APLICAȚII ÎN CHIMIA CUANTICĂ ȘI CINETICA CHIMICĂ

3.1. CALCULUL ORBITALILOR MOLECULARI

3.1.1. INTRODUCERE

Noțiunea fundamentală a chimiei este legătura chimică. Reacțiile compușilor chimici organici sau anorganici înseamnă *de facto* un proces (sau mai multe procese) în care se desfac și se refac legături chimice. Experimentatorul constată că în urma încercărilor sale practice, un compus și-a modificat structura, fie în concordanță cu intențiile sale, fie că evoluția procesului chimic a avut o direcție necunoscută.

Atunci cînd un chimist își planifică o lucrare experimentală, el trebuie să schițeze o schemă principală, care este cea mai probabilă linie evolutivă a experimentului său, în același timp avînd în vedere faptul că de multe ori chimia este „o analogie care nu se verifică", chimistul trebuie să ia în considerare și alte variante posibile de evoluție a sistemului chimic. Această schemă de lucru este desigur ideală, în realitate în această strategie este necesară alocarea unui loc și necunoscutului, adică unor reacții noi, fără analogie în materialul faptic pe care știința chimică îl posedă pînă la acea dată.

Orice chimist prevede, explică și înțelege obiectul preocupărilor sale profesionale prin prisma noțiunii de *structură electronică*. Cu cît cunoaștem mai mult despre structura electronică a compușilor chimici, cu atît vor exista mai puține „pete albe" în chimie.

Instrumentul de lucru, cu ajutorul căruia se obțin informațiile în legătură cu comportarea electronilor în compușii chimici, atomi sau molecule, este teoretic, constituind obiectul *chimiei cuantice*. Rezultatele teoretice, obținute prin calcule, se corelează cu unele procese dinamice, adică reacții chimice, sau cu rezultatele unor procese statice (procese care nu implică reacții chimice) ca spectroscopia fotoelectronică, de ultraviolet (UV), rezonanța magnetică nucleară (RMN), rezonanța electronică de spin (RES), determinările de moment de dipol sau chiar cu spectroscopia de infraroșu (IR). Așadar, există un front foarte larg de-a lungul căruia se pot stabili corespondențe între un instrument teoretic (chimia cuantică) și numeroase tehnici experimentale, în scopul de a cunoaște mai bine structura și comportarea compușilor chimici.

Scanned with OKEN Scanner

Avînd în vedere importanţa deosebită pe care o prezintă teoria structurii electronice în educaţia modernă a oricărui chimist experimentator, acest capitol va fi dedicat unor aspecte de calcul a orbitalilor moleculari. Chimia cuantică a sistemelor moleculare are la bază cîteva ecuaţii fundamentale, a căror aplicare este posibilă numai dacă se introduc anumite aproximaţii, evident nu la întîmplare, ci justificate. Din aceste motive procedeele de calcul existente în literatură sînt foarte numeroase, în plus numărul lor creşte rapid. Descrierea în detaliu, exhaustivă, a unor astfel de procedee ar fi anevoioasă şi poate inutilă, de aceea credem că prezentarea numai a unor metode, considerate a fi de largă circulaţie, este suficientă pentru a ne introduce în „atmosfera" unor programe.

Programele folosite astăzi, în mod obişnuit, în calcule de orbitali moleculari au un grad de complexitate ridicat. De exemplu, programul care are la bază teoria Hückel extinsă (adică include toţi electronii de valenţă) este alcătuit din 300—800 cartele, CNDO într-o variantă simplificată conţine 1200—1500 cartele, în timp ce un program scris pe baza metodei *ab-initio* conţine aproximativ 10 000 de cartele. Din aceste motive, scrierea integrală a unui program de calcul a orbitalilor moleculari, chiar în variante simplificate, este o sarcină dificilă, dar nu irealizabilă. Scrierea unui program cere în primul rînd cunoaşterea perfectă a limbajului de programare, a obiectului care va fi programat şi desigur o experienţă în folosirea calculatoarelor. Oricare cititor, care a parcurs cu atenţie materialul prezentat pînă acum, şi l-a însuşit şi aplicat, scriind programe, este capabil să abordeze elementele ce vor fi prezentate în acest ultim capitol.

Deşi, complexitatea programelor ce urmează a fi folosite s-a accentuat, există totuşi numeroase trasee, a căror parcurgere permite abordarea treptată, din ce în ce mai complexă, a unor astfel de probleme dificile.

Ideal ar fi ca la început să încercăm să utilizăm programe care ne interesează în mod deosebit pentru studiul unor anumiţi compuşi chimici.* Desigur, nu orice compus chimic poate con-

---

* În vederea facilitării răspîndirii programelor de calcul, s-a creat în 1963 QCPE (*Quantum Chemistry Program Exchange*) la Bloomington, Indiana 47401, S.U.A., cu o filială în Europa Deutsches Rechenzentrum, 6100 Darmstadt, Rheinstrasse 75, R.F.G., de unde se pot obţine la cerere, contra cost, programe-sursă, sau gratuit, liste ale programelor conform catalogului editat al organizaţiei. Alte surse de programe: *Computer Programs for Chemistry*, Delos F. Detar, Editor, W. A. Benjamin, Inc., New York, N. Y., vol. I (1968), vol II (1969), vol. III (1969).

Scanned with OKEN Scanner

stitui obiectul de studiu din punctul de vedere al chimiei cuantice. Dacă ne interesează sisteme chimice conţinînd numai schelete tip σ, vom utiliza metode care iau în considerare toţi electronii de valenţă (EH, CNDO, MINDO etc). În cazul în care sistemul este plan şi posedă electroni π, putem utiliza metode specifice acestor sisteme ca: Hückel, PPP-SCF etc., sau oricare din metodele enumerate în cazul sistemelor σ.

Utilizarea programului, cu un minim de recomandări referitor la modul de alcătuire a datelor necesare efectuării calculelor, cu o perseverenţă indusă de interesul nostru pentru o problemă chimică, va face ca treptat să dispară prejudecăţi, timiditate etc., care pentru foarte mulţi constituie o frînă în folosirea unei tehnici de calcul, aparent copleşitoare, cum ar fi aceea oferită de un calculator. Dar oare la început este vital-necesar să ştim cum operează programul? Cred că în mod curent în viaţa cotidiană am lucrat şi lucrăm cu obiecte despre care chiar dacă la început nu ştim *cum* lucrează, ştim însă *cum să* lucrăm cu ele.

Un program de calcul al orbitalilor moleculari are trei părţi distincte, dintre care două sînt comune tuturor metodelor.

Prima parte, diferită de la un program la altul, o constituie segmentul de alcătuire a hamiltonianului (v. pag. 132), construit în funcţie de aproximaţiile metodei respective. A doua parte, diagonalizarea hamiltonianului şi a treia parte, prelucrarea vectorilor proprii şi a valorilor proprii în mărimi ca ordine de legătură, sarcini electronice, momente de dipol etc. sînt comune tuturor programelor.

Deci, atunci cînd dorim să efectuăm modificări într-un program, în vederea utilizării unei metode noi de calcul, vom opera în mod substanţial în segmentul de construire a hamiltonianului. Aşadar, am introdus al doilea pas, pasul în care nu ne mai mulţumeşte metoda utilizată şi vrem de aceea să o modificăm. În acest moment ştim să utilizăm programe şi ştim să scriem programe!

Înainte de a scrie orice program, principiul de bază al programatorului trebue să fie cunoaşterea perfectă a problemei de programat. Din acest motiv vom încerca, la începutul fiecărei probleme care urmează a fi exemplificată printr-un program, să prezentăm un cadru teoretic general care circumscrie problema ce urmează a fi programată, precum şi detaliile me-

Scanned with OKEN Scanner

todei respective, întocmai cum am procedat şi în cazul exemplificărilor prezentate pînă acum.

Orice *moleculă* poate fi considerată ca un conglomerat de particule (nuclee şi electroni) între care există interacţii guvernate de diverse legi ale fizicii. Ecuaţia fundamentală a mecanicii cuantice, ecuaţia lui Schrödinger, aplicată unui sistem chimic, ar trebui să ne ofere suficiente informaţii în legătură cu structura sa electronică. Deşi problema se reduce la o simplă aplicare, trebuie avut în vedere că obiectul de studiu al chimiei cuprinde o gamă extraordinară de compuşi situaţi între două extreme: una constituită din sistemele atomice (între care cel mai simplu e hidrogenul), iar cealaltă din sisteme constituite din sute de mii de atomi, adică compuşii macromoleculari. Aplicarea ecuaţiilor fundamentale ale chimiei cuantice, chiar în cazul unor sisteme simple, ridică probleme foarte complexe de teorie matematică cît şi de calcul numeric, încît este necesară introducerea unor metode aproximative. Cuvîntul de „aproximaţie" ne îndeamnă la o anumită circumspecţie. Cîtă încredere se poate investi în astfel de calcule? Totuşi, să nu uităm că metode aproximative de calcul există în toate ştiinţele fizice şi ori de cîte ori ne referim la aspecte aplicate vom observa că astfel de metode aproximative constituie majoritatea, în timp ce metodele exacte reprezintă minoritatea. Dar nu precedentele sînt cele mai bune argumente, mult mai elocvente sînt argumentele „chimice". Astăzi, după peste 40 ani de experienţă (şi în mod special în ultimii 15 ani, odată cu dezvoltarea calculatoarelor), aportul adus de chimia cuantică chimiei ca ştiinţă experimentală a fost substanţial. Chimiştii sînt foarte sensibili la nou, mai ales cînd acesta le sistematizează cunoştinţele, cînd fenomenele chimice sînt mai bine înţelese, cînd ei pot să prevadă reacţii noi. Să nu uităm că multe industrii de astăzi au la bază şi cercetări stimulate de aplicarea în chimie a mecanicii cuantice. Pentru a concretiza, am menţionat doar două exemple care au constituit şi mai constituie momente de cotitură, de impact în chimia organică. Este interesant de menţionat că aceste exemple au baze de calcul care din punct de vedere teoretic sînt clasate naive de către teoreticienii timpurilor respective.

Astfel în 1931, E. Hückel[1] considera că pentru a avea proprietăţi aromatice, un sistem polienic, ciclic şi plan cu conjugare neîntreruptă trebuie să posede un număr de $(4n + 2)$ electroni $(n = 0,1,2 \ldots)$. Astăzi deşi se cunoaşte suficient de bine domeniul de valabilitate a acestei reguli, interesul pentru chimia caracterului aromatic, pseudoaromatic şi antiaromatic (no-

Scanned with OKEN Scanner

țiuni generate de această regulă) este încă mare. Chimia acestui domeniu, avînd numeroase aplicații industriale, a cunoscut de atunci o dezvoltare impresionantă.

În 1965, R. B. Woodward și R. Hoffmann publicau o suită de reguli referitoare la reacții electrociclice, sigmatropice, reacții de cicloadiție, reguli bazate pe principiul conservării simetriei orbitalilor[2]. Aceste lucrări au impulsionat un număr impresionant de cercetări științifice. Astăzi chimistul organician înțelege mult mai bine mersul acestor reacții chimice, condițiile experimentale de realizare a lor potrivit unui anumit curs steric.

Se poate spune, fără a fi o falsă încurajare, că deși sînt aproximative, metodele de calcul ale structurii electronice, exploatate rațional, în mîna unui experimentator talentat, vor genera soluții practice, vor însemna o mai bună înțelegere a fenomenologiei chimice, ceea ce va avea consecințe deosebite asupra progresului chimiei.

3.1.2. ASPECTE GENERALE ALE METODELOR DE CALCUL A ORBITALILOR MOLECULARI

Înainte de a discuta în mod concret diversele metode de calcul, vom formula un cadru general, în care se va arăta obiectul chimiei cuantice și modul de rezolvare a ecuațiilor fundamentale, cu scopul de a facilita descrierea metodelor concrete de calcul al orbitalilor moleculari. În fond, aceste metode constituie diverse nivele de aproximație, care se referă în special la calculul unor integrale sau înlocuirea lor cu parametrii semiempirici.

O stare staționară „i" se descrie de o ecuație Schrödinger de forma:

$$H\psi_i = E_i\psi_i. \tag{3.1.1}$$

În această ecuație, H este operatorul lui Hamilton al sistemului; ψ_i — funcția de undă care descrie starea staționară „i" și care depinde de coordonatele geometrice și de spin al întregului sistem de particule (nuclee și electroni); E_i — energia asociată stării electronice ψ_i. Explicitarea lui H și rezolvarea ecuației lui Schrödinger reclamă introducerea unor aproximații.

Aproximația I (cunoscută sub numele de aproximația Born-Oppenheimer). Se consideră că mișcarea nucleelor este separabilă de mișcarea electronilor. În acest caz, hamiltonianul sistemului H, poate fi considerat ca fiind alcătuit din termeni care repre-

Scanned with OKEN Scanner

zintă energia cinetică a fiecărui electron $(-1/2\,\nabla_i^2)$ și termeni de energie potențială rezultați din interacțiile nucleu-nucleu, electron-nucleu și electron-electron. Formularea lui H în unități atomice* este următoarea:

$$H = \sum_i (-1/2 \cdot \nabla_i^2) + \sum_M \sum_{N>M} \frac{Z_M \cdot Z_N}{r_{MN}} - \sum_M \sum_i \frac{Z_M}{r_i} + \\ + \sum_i \sum_{j>i} \frac{1}{r_{ij}}, \qquad (3.1.2)$$

în care Z_N și Z_M sînt sarcinile nucleare ale nucleelor M și N; r—distanța; i și j—indici cu care se definesc electronii.

Aproximația II (cunoscută sub numele de aproximația orbitală). Rezolvarea ecuației (3.1.1) este substanțial simplificată dacă funcția multielectronică ψ_i (1,2,...) ar fi factorizată în funcții monoelectronice de tipul $\varphi(1)\ \varphi(2)\ \ldots$, în care fiecare funcție este dependentă numai de coordonatele geometrice și de spinul unui singur electron. Într-o astfel de formulare, ar însemna că putem spune care electron ocupă un anumit spinorbital (reprezentat prin φ). În realitate, o astfel de atribuire este imposibilă. Potrivit principiului lui Pauli, în forma sa cea mai generală, o funcție de undă trebuie să fie antisimetrică (adică să-și modifice semnul) ori de cîte ori se schimbă coordonatele a doi electroni. Această condiție este realizată numai dacă se formulează funcția de undă sub forma unui determinant Slater

$$\psi_i(1,2,\ldots n) = |\varphi_1(1)\ \varphi_2(2)\ \ldots\ \varphi_n(n)| \qquad (3.1.3)$$

care conține toate permutările posibile de coordonate electronice între toți spinorbitalii φ_i, fiecare permutare avînd semnul corespunzător. În acest fel, atribuirea unui anumit electron unui anumit spinorbital nu mai este posibilă.

Conform proprietăților determinanților, schimbarea a două rînduri (sau coloane) între ele are drept consecință modificarea semnului determinantului respectiv, cu alte cuvinte se respectă automat principiul lui Pauli (v. supra).

---

* Unitatea de masă este masa de repaus a electronului ($m_e = 9{,}1091 \times 10^{-28}$ g), unitatea de lungime este prima rază Bohr la cazul atomului de hidrogen, calculată, înlocuind masa redusă cu m_e ($0{,}52917 \times 10^{-8}$ cm), unitatea de energie = 27,210 eV, unitatea de timp este egală cu $2{,}42 \times 10^{-17}$ s, adică timpul necesar unui electron să parcurgă în prima orbită Bohr distanța de 1 u.a.

Scanned with OKEN Scanner

Aproximația III. Fiecare spinorbital molecular φ_i se consideră ca o combinație liniară de orbitali atomici*, multiplicați prin funcția de spin s_i (care poate fi α sau β):

$$\varphi_i = \left(\sum_{j=1}^{n} C_{ij} \Phi_j \right) \cdot s_i. \qquad (3.1.4.)$$

Care sînt orbitalii atomici care prezintă interes în calculul orbitalilor moleculari? Alegerea orbitalilor atomici este sugerată de intuiția chimică; în procesele chimice participă numai electronii de valență. În cazul sistemelor conjugate, proprietățile chimice sînt determinate de electronii, π și prin urmare orbitalii atomici care pot fi luați în considerare sînt numai orbitalii P_Z.

Astfel, modul de alegere a orbitalilor atomici reprezintă un criteriu de clasificare a metodelor de calcul în:

a) metode de calcul al sistemelor π, în care orbitalii atomici sînt reprezentați doar prin orbitalii P_Z. Din această categorie fac parte metoda Hückel[3] (HMO) și metoda Pariser-Parr-Pople[4] (PPP);

b) metode de calcul în care se iau în considerare toți orbitalii de valență ai atomilor. Atomii din perioada a II-a au 4 orbitali de valență $(S + P_X + P_Y + P_Z)$, în timp ce elementele perioadei a III-a posedă 10 orbitali de valență

$$(S + P_X + P_Y + P_Z + d_{Z^2} + d_{X^2-Y^2} + d_{XY} + d_{XZ}).$$

Astfel de metode sînt metoda Hückel extinsă (EHM)[5], CNDO[6], INDO[6] și MINDO[7].

În practică, se utilizează diverse funcții pentru orbitalii atomici. În mod curent se utilizează funcțiile Slater[8]:

$$\begin{aligned}
\Phi_{1S} &= (\zeta_1^3/\pi)^{1/2} \exp(-\zeta_1 r) \\
\Phi_{2S} &= (\zeta_2^5/3\pi)^{1/2} \cdot r \cdot \exp(-\zeta_2 r) \\
\Phi_{2P_x} &= (\zeta_2^5/\pi)^{1/2} \cdot x \cdot \exp(-\zeta_2 r) \qquad (3.1.5) \\
\Phi_{2P_y} &= (\zeta_2^5/\pi)^{1/2} \cdot y \cdot \exp(-\zeta_2 r) \\
\Phi_{2P_z} &= (\zeta_2^5/\pi)^{1/2} \cdot z \cdot \exp(-\zeta_2 r)
\end{aligned}$$

unde ζ sînt parametrii care determină mărimea orbitalului.

---

* În literatură această aproximație este prescurtată LCAO, *Linear Combination of Atomic Orbitals.*

Scanned with OKEN Scanner

Calculele cu orbitalii Slater sînt laborioase, fiind mari consumătoare de timp în etapa de calcul a integralelor (v. exemple la pag. 160)

În zilele noastre, încep să fie tot mai mult preferate funcţiile Gauss[9] care sînt derivate din funcţiile Slater dacă se fac următoarele substituţii:

$$\exp(-r) \rightarrow \sum_{i=1}^{N} a \cdot \exp(-\alpha_{1i} r^2)$$

$$r \cdot \exp(-r) \rightarrow \sum_{i=1}^{N} a \cdot \exp(-\alpha_{2i} r^2)$$

$$x \cdot \exp(-r) \rightarrow \sum_{i=1} a_{2P_x} \cdot \exp(-\alpha_{2i} r^2) \qquad (3.1.6)$$

$$y \cdot \exp(-r) \rightarrow \sum_{i=1}^{N} a_{2P_y} \cdot \exp(-\alpha_{2i} r^2)$$

$$z \cdot \exp(-r) \rightarrow \sum_{i=1}^{N} a_{2P_z} \cdot \exp(-\alpha_{2i} r^2)$$

Coeficienţii a şi α sînt determinaţi cu ajutorul metodei lui Stewart[10].

Evaluarea integralelor care implică sume de funcţii Gauss este rapidă, rezultatele sînt bune atunci cînd se utilizează în calcule sume cu $N = 3,4,5$ şi chiar 6. Unicul „dezavantaj" al utilizării funcţiilor Gauss este faptul că programele cer memorie internă mare, din cauză că fiecare orbital este „simulat" de o sumă de funcţii Gauss.

În concluzie, pînă în acest moment am trecut în revistă cele trei aproximaţii de ordin teoretic, care permit simplificarea ecuaţiei lui Schrödinger la o formă rezolvabilă. În cele ce urmează vom prezenta ecuaţiile concrete, de lucru, pentru calculul orbitalilor moleculari, aşa cum au fost deduse de către Roothaan[11]. O parte din aceste metode, care utilizează aproximaţii bine testate, vor fi prezentate în cadrul acestui capitol.

Pentru rezolvarea ecuaţiei (3.1.1) este suficient să determinăm coeficienţii C_{ij} din ecuaţia (3.1.4). Energia sistemului, E, este funcţie de aceşti coeficienţi; vor prezenta interes numai acele valori ale lui C_{ij} care conferă lui E o valoare minimă.

Scanned with OKEN Scanner

Minimizarea lui E în raport cu toţi coeficienţii C_{ij} generează un set de ecuaţii liniare omogene* :

$$\sum_j (F_{ij} - \varepsilon_k \cdot S_{ij}) \cdot C_{ik} = 0, \tag{3.1.7}$$

care are soluţii nebanale numai dacă determinantul secular se anulează :

$$|F_{ij} - \varepsilon_k \cdot S_{ij}| = 0, \tag{3.1.8}$$

unde F_{ij} sînt elementele matricii lui Fock ; S_{ij} — integrala de acoperire $\int \varphi_i \varphi_j \, dv$; ε_k — energia corespunzătoare unei funcţii monoelectrice.

Cel mai important termen al matricii (3.1.8) este F_{ij}, care este convenabil a se formula ca o sumă :

$$F_{ij} = H_{ij} + G_{ij}, \tag{3.1.9}$$

unde H_{ij} este un termen independent de numărul de electroni care participă la completarea nivelelor moleculare şi este definit astfel :

$$H_{ij} = \int \Phi_i \left[-1/2 \, \nabla_i^2 - \sum_M \frac{Z_M}{r} \right] \Phi_j d\tau, \tag{3.1.10}$$

în timp ce G_{ij} este un termen dependent de numărul electronilor participanţi la ocuparea orbitalilor moleculari. Considerăm cazul în care numărul de electroni este par, adică sistemul electronic posedă strat închis (v. ecuaţia 3.1.14, în care apare coeficientul 2)

$$G_{ij} = \sum_{k,m} p_{km} [\langle ij \mid km \rangle - \frac{1}{2} \langle ik \mid jm \rangle], \tag{3.1.11}$$

în care :

$$\langle ij|km \rangle = \int\int \Phi_i(1)\Phi_j(2) \frac{1}{r_{12}} \Phi_k(1) \cdot \Phi_m(2) \, d\tau \tag{3.1.12}$$

$$\langle ik|jm \rangle = \int\int \Phi_i(1) \cdot \Phi_k(2) \frac{1}{r_{12}} \Phi_j(1) \cdot \Phi_m(2) \, d\tau \tag{3.1.13}$$

---

* Folosirea indicilor i, j, k (ec.4 şi ec. 7) nu trebuie să creeze confuzii, deoarece important este să avem în vedere că şi coeficienţii C alcătuiesc o matrice, din care fac parte atît C_{ij} cît şi C_{ik}.

Scanned with OKEN Scanner

și P_{km} sînt elementele matricii ordinelor de legătură:

$$P_{km} = 2 \sum_{i=1}^{OCUP} C_{ik} C_{im}, \tag{3.1.14}$$

în care suma se calculează peste toți orbitalii moleculari ocupați.

Cititorul care a parcurs atent aceste rînduri va avea desigur o surpriză cînd va constata că, în calculul elementelor de matrice F_{ij}, trebuie să cunoaștem de fapt necunoscutele C_{ij}. În acest caz, rezolvarea nu se poate face într-o singură etapă, ea necesită mai multe etape (iterații).

Deoarece, inițial $C_{ij} = 0$, înseamnă că (3.1.9) se reduce la:

$$F_{ij} = H_{ij} \tag{3.1.15}$$

și deci determinantul secular devine:

$$|H_{ij} - \varepsilon_k S_{ij}| = 0. \tag{3.1.16}$$

Diagonalizarea acestei ecuații (3.1.16), cu metoda Jacobi sau Givens (v. pag. 109) conduce la un set de coeficienți, calculați *în aproximația zero*. Cu acești coeficienți se poate calcula p_{km}, care servesc la aflarea lui F_{ij} (aproximația de ordinul I), matrice care permite să se calculeze un nou set de coeficienți. Criteriul potrivit căruia stabilim în ce moment ne declarăm mulțumiți cu rezultatele, adică cu valorile ε_k (valorile proprii sau nivele energetice) și C_{ij} (vectorii proprii sau coeficienții din funcțiile moleculare) poate fi ales în mai multe moduri:

1) Criteriu energetic: diferența dintre două valori ale energiei sistemului, obținute în iterații consecutive, să fie mai mică decît o constantă arbitrară, de exemplu 10^{-3}.

2) Criteriul convergenței coeficienților funcțiilor moleculare: diferența între coeficienții C_{ij}, calculați în două iterații consecutive, să fie mai mică decît o constantă arbitrară, de valoare foarte mică, de exemplu 10^{-4}.

3) Criteriile (1) și (2) să fie îndeplinite simultan.

Dacă notăm prin F matricea Fock (constituită din elemente F_{ij}) și prin C matricea coeficienților C_{ij}, schema iterațiilor succesive poate fi formulată astfel:

Se dă	Se calculează	Se obține
1. $C = 0$	F^0	C^0
2. C^0	F^1	C^1
3. C^1	F^2	C^2
⋮		
n. C^{n-1}	F^n	C^n

Scanned with OKEN Scanner

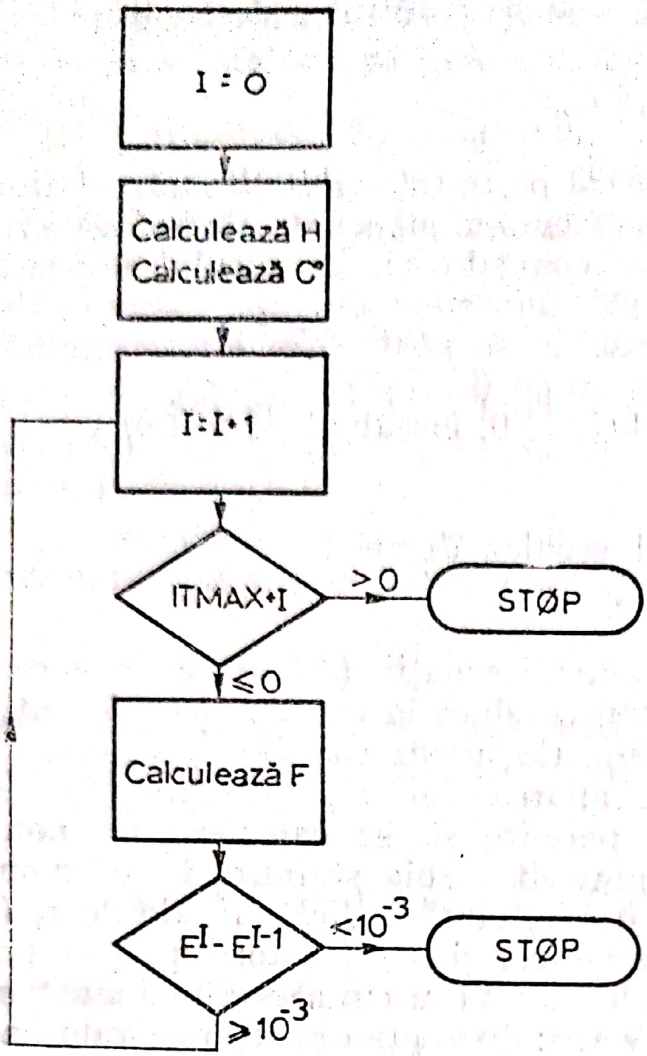


Fig. 3.1

Uneori acest calcul iterativ nu are convergenţă, diferenţele $E = E^n - E^{n-1}$ sau/şi $C = C^n - C^{n-1}$ nu devin, chiar pentru valori mari ale lui n, mai mici sau egale cu numărul mic, ales arbitrar (adică 10^{-3} şi respectiv 10^{-4}).

În alte cazuri nu se stabileşte un criteriu de convergenţă, ci se preferă să se efectueze un număr dat de iteraţii, de exemplu $n = 10$, considerînd în mod obişnuit, dacă sistemul are convergenţă, că acest număr de iteraţii este suficient pentru cerinţele obiectivului urmărit.

Deoarece calculul prin iteraţii este o tehnică fundamentală în calculul funcţiilor de undă moleculare, prin care se „realizează" un cîmp autocoerent*, vom schiţa o secvenţă dintr-un

* Nomenclatura anglo-saxonă foloseşte terminologia de *Self Consistent Field*, sau prescurtat SCF.

program care foloseşte criteriul energetic drept criteriu de convergenţă. Variabila I este aceea în care se stochează numărul de iteraţii. Pentru a evita situaţiile de divergenţă sau convergenţă lentă, cînd numărul de iteraţii poate să aibă valori foarte mari, şi deci calculul ar consuma inutil timp, blocînd nejustificat calculatorul, putem să introducem o valoare maximă pentru numărul de iteraţii, la discreţia celui care rulează programul, denumit ITMAX (prescurtare de la *it*eraţii *max*ime).

3.1.3. METODA HÜCKEL DE CALCUL A ORBITALILOR MOLECULARI[1,12]

Deşi metoda Hückel este cadrul celor mai drastice aproximaţii care se operează în determinantul secular al lui Roothaan, totuşi, poate incredibil, ea a făcut epocă şi poate că prezintă, în anumite situaţii, un oarecare interes şi astăzi. Cu toate că teoreticienii o ignoră complet, cărţile o menţionează, dar nu pentru valoarea ei istorică, ci pentru faptul că reprezintă un foarte valoros material didactic. Pentru nespecialişti, momentul Hückel, în străduinţa de a cunoaşte tainele chimiei cuantice, este o etapă hotărîtoare. Cu ajutorul ei se exemplifică numeroase noţiuni fundamentale, cum ar fi nivele energetice, funcţii de undă moleculare, densitate electronică, ordin de legătură, moment de dipol, tranziţii electronice etc. Cu toate că literatura este copleşită de metodele sofisticate de calcul a structurii electronice, mai există astăzi suficiente probleme de chimie organică care pot fi înţelese şi interpretate utilizînd metoda Hückel.

Interpretarea rezultatelor reclamă prudenţă, deseori e mai inteligent să transpunem rezultatele cantitative în concluzii calitative, cu alte cuvinte e mai înţelept în anumite cazuri să manevrăm cuvinte şi nu cifre.

În partea de introducere la acest capitol am menţionat că, în calculul energiilor şi coeficienţilor funcţiilor de undă, etapa cea mai importantă este etapa de diagonalizare a matricii Hückel. Deoarece în cap. 2 s-au descris tehnici de diagonalizare şi modul de inserare a acestora în programe, în rîndurile care urmează ne interesează numai modul de construcţie a matricii Hückel, deci implicit datele necesare pentru rezolvarea problemei şi modul de prelucrare a coeficienţilor funcţiilor de undă moleculare în vederea obţinerii unor informaţii referitoare la distribuţia electronilor π în moleculele conjugate, concretizate în calculul sarcinii electronice şi indicelui de legătură.

Scanned with OKEN Scanner

Elementele de matrice se aleg în felul următor:

— elementele diagonale

$$F_{ii} = \alpha_i \tag{3.1.17}$$

— elementele nediagonale

$$F_{ij} = \begin{cases} \beta_{ij} \text{ dacă } r \text{ și } s \text{ sînt vecini} \\ 0 \text{ dacă } r \text{ și } s \text{ sînt vecini} \end{cases} \tag{3.1.18}$$

— acoperirile se neglijează:

$$S_{ij} = 0 \tag{3.1.19}$$

și deci

$$|F_{ij} - \varepsilon \cdot 1| = 0 \tag{3.1.20}$$

În cadrul acestei metode se introduc doi parametri, α — denumit integrala coulombiană — și β, denumit integrala de rezonanță. O matrice, construită conform condițiilor (3.1.17), (3.1.18) și (3.1.19), reflectă, în fond, topologia sistemului π.

Există elemente F_{ij}, numai dacă între „i" și „j" există o legătură chimică.

În cazul cînd din sistemul π fac parte unul sau mai mulți heteroatomi, integralele coulombiene și de rezonanță ale acestora se calculează, raportate la cele ale carbonului luate ca standard, adică cu ajutorul unor formule de tipul:

$$F_{ii} = \alpha \pm m\beta \tag{3.1.21}$$

$$F_{ij} = k\beta. \tag{3.1.22}$$

Cîteva valori pentru m și k sînt date în tabelul 3.1.

TABELUL 3.1.

COEFICIENȚII INTEGRALELOR COULOMBIENE ȘI DE REZONANȚĂ PENTRU ATOMII PERIOADEI A II-a

Atom	m	k
C	0	1.
N	1.5	1.
N	0.5	0.8
O	2.0	1.
F	3.0	0.7

Scanned with OKEN Scanner

Metoda Hückel poate fi modificată pentru a deveni o metodă autocoerentă[13]. Desigur că o astfel de modificare reprezintă o îmbunătăţire. Termenul diagonal H_{ii} se consideră dependent de sarcina electronică existentă la nucleul respectiv:

$$H_{ii}^{n} = H_{ii}^{0} + \omega(1 - P_{ii}^{n-1}), \tag{3.1.23}$$

unde H_{ii}^{0} este valoarea iniţială a termenului diagonal, P_{ii}^{n-1} — sarcina electronică pe nucleul „i" în iteraţia $n-1$, iar ω — un parametru empiric, de obicei avînd valoarea 1,4. Dacă nucleul conţine doi electroni participanţi la sistemul π, formula de calcul devine:

$$H_{ii}^{n} = H_{ii}^{0} + \omega\,(2 - P_{ii}^{n-1}). \tag{3.1.24}$$

Simultan cu modificările termenilor diagonali se poate efectua şi modificarea termenilor nediagonali.

Termenul H_{ij} este funcţie de distanţa dintre „i" şi „j", R_{ij}, şi aceasta, la rîndul ei, este funcţie de ordinul de legătură între centrele „i" şi „j". Conform relaţiei lui Longuet-Higgins şi Salem[14a], termenul nediagonal se poate formula:

$$H_{ij}^{n} = H_{ij}^{0} \exp((-R_{ij} - 1.397) / 0.3106), \tag{3.1.25}$$

unde R_{ij} se poate calcula cu relaţia lui Coulson[14b]:

$$R_{ij} = 1.517 - 0.18\ P_{ij}^{n-1}, \tag{3.1.26}$$

unde P_{ij}^{n-1} este ordinul de legătură între centrele „i" şi „j", care este calculat cu ajutorul coeficienţilor funcţiilor de undă moleculare ocupate cu m_k electroni (2 sau 1):

$$P_{ij}^{n-1} = \sum_{k=1}^{OCUP} m_k C_{ki}^{n-1} C_{kj}^{n-1}. \tag{3.1.27}$$

Dacă $i = j$, P_{ii} reprezintă sarcina electronică la centrul „i":

$$P_{ii}^{n-1} = \sum_{k=1}^{OCUP} m_k \cdot C_{ki}^{2}. \tag{3.1.28}$$

Ecuaţiile (3.1.23) — (3.1.28) alcătuiesc structura pe care va fi dezvoltat un program de calcul al orbitalilor moleculari prin metoda ω.

Programul constă dintr-un program principal şi subrutine pentru calculul *or*dinului de *leg*ătură (ORLEG), a *sarci*nii elec-

tronice (SARCN), *ener*giei *tot*ale (ENTOT), diagonalizare (v. Jacobi, cap. 2) şi *ti*părirea (scrierea) *mat*ricelor (TIMAT).

Programul principal este utilizat pentru:

1. citirea datelor necesare calculului
2. scrierea rezultatelor
3. calculul elementelor de matrice H_{ii} şi H_{ij}
4. calculul iterativ

Pregătirea datelor iniţiale.

Cartela I cuprinde: (format (6(I2,1X)))

NØCAZ — numărul cazului
NATØM — numărul atomilor
NELEC — numărul de electroni
ITMAX — numărul de iteraţii maxime
LLEG — lungimea legăturilor (da = 1, nu = 0)
MLEC — indice de modificare (da = 1, nu = 0) a lungimii legăturii în timpul iteraţiilor.

Cartela II cuprinde titlul problemei (format 18A4).

Cartela III şi următoarele, în cazul în care drept date iniţiale se folosesc lungimi de legătură (LLEG = 1), conţin (format 2I2, F10.5)

I. J, RLEG (I,J)

unde I şi J sînt atomii între care există legătura de lungime RLEG (I,J).

Pentru fiecare legătură existentă în moleculă apare o astfel de cartelă. Încheierea datelor privind legăturile din sistem se face cu o cartelă neperforată, care echivalează cu I = J = = RLEG (I,J) = 0.

Dacă datele iniţiale sînt integralele de rezonanţă (LLEG = = 0) atunci cartela III şi următoarele conţin

I, J, BETA

unde BETA este valoarea integralei de rezonanţă dintre orbitalii P_z plasaţi la atomii I şi J. În acest caz, dacă molecula conţine heteroatomi, vor apărea şi cartele cu I = J şi BETA = = m (vezi pag. 136), introducîndu-se astfel integrala coulombiană a heteroatomului. Încheierea datelor va fi din nou semnalată printr-o cartelă neperforată.

În cazul în care sistemul conţine atomi care participă cu doi electroni la sistemul π (adică sisteme heteroatomice), această informaţie este introdusă prin intermediul unor cartele care conţin:

I, I, RLEG (I,I)

Scanned with OKEN Scanner

unde I este numărul heteroatomului iar RLEG (I,J) = 2. semnalează participarea a doi electroni de la centrul I.

Stocarea informației în diagonala matricei RLEG este permisă, deoarece în mod normal acești termeni sînt nuli, întrucît lungimi de legătură nu pot exista decît între atomi diferiți. Și aceste date se încheie printr-o cartelă neperforată.

În cazul în care dorim să calculăm, la o singură rulare a programului, mai multe cazuri (molecule) este suficient să punem unul după altul seturile de date privind fiecare moleculă.

Pentru a provoca întreruperea calculelor, după ultimul set de cartele de date trebuie pusă o cartelă neperforată. În momentul în care va fi citită, această cartelă (care de fapt înseamnă NØCAZ = 0, NATØM = NELEC = etc. = 0) va determina transferul la instrucțiunea STØP.

Ca prim exemplu de date de intrare (NØCAZ = 1) să considerăm butadiena. Sistemul de numerotație va fi:

```
C – C – C – C
1   2   3   4
```

Vom considera, ca o primă variantă, integralele de rezonanță drept date de intrare (LLEG = 0). Stabilind MLEG = 1, înseamnă că dorim ca în timpul iterațiilor să se modifice și termenul nediagonal. Butadiena are un număr de 4 atomi, deci NATØM = 4; un număr de patru electroni π, deci NELEC = = 4. Numărul iterațiilor maxime ITMAX = 20.

Deci, setul de cartele cu date va fi*:

```
01 04 04 20 00 01
BUTADIENA VARIANTA I LLEG = 0, MLEG = 1
0102 1.
0203 1.
0304 1.
0 0
0 0
```

În varianta a doua (NØCAZ = 2), considerăm ca date de inițiale pentru calculul elementelor nediagonale lungimea

---

* Cartelele neperforate sînt menționate prin doi de zero.

Scanned with OKEN Scanner

legăturilor (LLEG = 1) $C_1 - C_2 = 1.337$Å, $C_2 - C_3 = 1.483$, și $C_3 - C_4 = 1.337$. Datele de input vor fi:

```
02 04 04 20 01 01
BUTADIENA VARIANTA II CU LUNGIMEA LEG. PT. CALCULUL
TERMENULUI NØNDIAGØNAL
01  02  1.337
02  03  1.483
03  04  1.337
00
00
```

Vom exemplifica modul de tratare a sistemelor de heteroatomi cu ajutorul a trei exemple.

NØCAZ = 3 – acroleina:

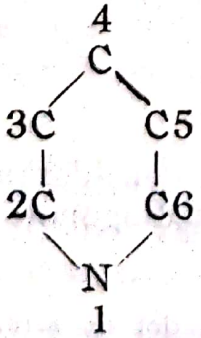
C—C—C—O
1 2 3 4

Din punct de vedere electronic, acroleina are 4 electroni π, întocmai ca butadiena, însă atomul 4 fiind oxigen, înseamnă că integrala coulombiană și de rezonanță se calculează după formulele (3.1.21) și (3.1.22). Astfel $m_0 = 2$ și $k_{C-O} = 1$.

Pentru acest caz LLEG – MLEG = 0; nu vom folosi ca input lungimi de legătură, iar termenul nediagonal nu va fi modificat în timpul calculelor.

```
03 04 04 20 00 00
ACRØLEINA, ATØMUL DE ØXIGEN ESTE NØ.4 cu M = 2 și K = 1
01 02  1.
02 03  1.
03 04  1.
04 04  2.
00
00
```

NØCAZ = 4, piridina



Pentru atomul de N s-a ales $m = 0.5$ şi $k_{CN} = 0.8$, numărul de atomi NATØM = 6, numărul de electroni este NELEG = = 6

```
04 06 06 20 00 00
PIRIDINA (ATØMUL DE AZØT ARE NØ.1, M = 0.5, N = 0.8)
01 01   0.5
01 02   0.8
01 06   0.8
02 03
03 04
04 05
05 06
00
00
```

NØCAZ = 5, pirolul

Este un exemplu care conţine un heteroatom pentru care am ales $m = 1.5$ şi $k = 1$ care posedă doi electroni π, ceea ce va fi menţionat în datele de input. Şi în acest caz considerăm LLEG = 0 şi MLEG = 0

```
05 05 06 20 00 00
PIRØLUL (ATØMUL DE AZØT ESTE NØ.1, M = 1.5, N = 1)
01 01   1.5
01 02   1.
01 05   1.
02 03
03 04
04 05
00
01 01 2
00
```

Citirea matricei de legături $i - j$ trebuie să se facă în aşa fel ca indicele j ($j > i$) să ia toate valorile, conform schemei de legătură stabilite pe baza structurii moleculei.

Scanned with OKEN Scanner

Astfel este corectă următoarea formulare:

```
01 01
01 02
01 06
 .
 .
```

şi nu

```
01 01
01 02
06 01
 .
 .
```

Descrierea programului

Există două părţi distincte în programul principal. Prima parte este destinată citirii datelor şi construcţiei matricei iniţiale Hückel, în timp ce în partea a doua se face calculul iterativ.

În ceea ce priveşte „subtilităţile" din prima parte a programului, am menţiona modul de citire extrem de simplu al elementelor de matrice. Să urmărim, de exemplu, modul de construire a matricei Hückel iniţiale HØ: elementele matricei se iniţializează în ciclări biindexate:

```
      DØ 3 I = 1, NATØM
      DØ 3 J = 1, NATØM
      HØ (I,J) = 0.0
    3 CØNTINUE
```

obţinîndu-se astfel:

$$H\text{Ø} = \begin{pmatrix} 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \end{pmatrix}$$

În continuare, elementele diferite de zero se citesc în conformitate cu topologia moleculei:

— ca lungimi de legătură din care apoi se calculează elementele de matrice utilizînd formulele (3.1.25-26);

— direct prin valorile lui BETA.

Conform numerotării, următoarele elemente sînt diferite de zero (vezi notaţiile pentru acroleină):

I	*J*	*BETA*		
1	2	1.		HØ(1,2) = 1.
2	3	1.	ceea ce este	HØ(2,3) = 1.
3	4	1.	echivalent cu	HØ(3,4) = 1.
4	4	2		HØ(4,4) = 2.

Deci:

$$HØ = \begin{pmatrix} 0. & 1. & 0. & 0. \\ 1. & 0. & 1. & 0. \\ 0. & 1. & 0. & 1. \\ 0. & 0. & 1. & 2. \end{pmatrix}$$

Sfîrşitul operaţiei de citire se semnalează cu o cartelă neperforată (ceea ce echivalează cu I = 0):

```
11 READ(IN,102)I,J,BETA
   HØ (I,J) = BETA
   IF(I)14,14,11
14 CØNTINUE
```

Atenţie: J > I

Diagonalizarea acestei matrice prin metoda Jacobi, CALL JACØBI(HØ, NATØM,U), permite obţinerea matricei vectorilor proprii, cu care se calculează matricea ordinelor de legătură, CALL ØRLEG(U, NATØM, NELEC,P).

Partea a doua a programului principal efectuează calculul iterativ, începînd cu instrucţiunea 3000. Calculul iterativ are două variabile: TEST, care reprezintă diferenţa dintre *ener*gia *ante*rioară (ENANT) şi energia actuală (ENERG), şi ITER, contorul de iteraţii. TEST se compară cu 0.001, adică:

$TEST = |Energie^{ITER+1} - Energie^{ITER}| \leqslant 0.001$ (unităţi β), ceea ce înseamnă realizarea convergenţei în conformitate cu opţiunea noastră, şi urmează tipărirea rezultatelor finale. Dacă TEST nu îndeplineşte condiţia de convergenţă (adică este mai mare decît 0.001), se compară mărimea contorului ITER cu ITMAX. Dacă ITMAX − ITER < 0 calculul este abandonat, sistemul fie nu are convergenţă, fie necesită prea multe iteraţii, şi se trece la problema următoare (GØ TØ 1), dacă există. Dacă ITMAX este încă mai mare decît ITER, se creşte contorul de iteraţii cu unu, ITER = ITER + 1, şi se începe

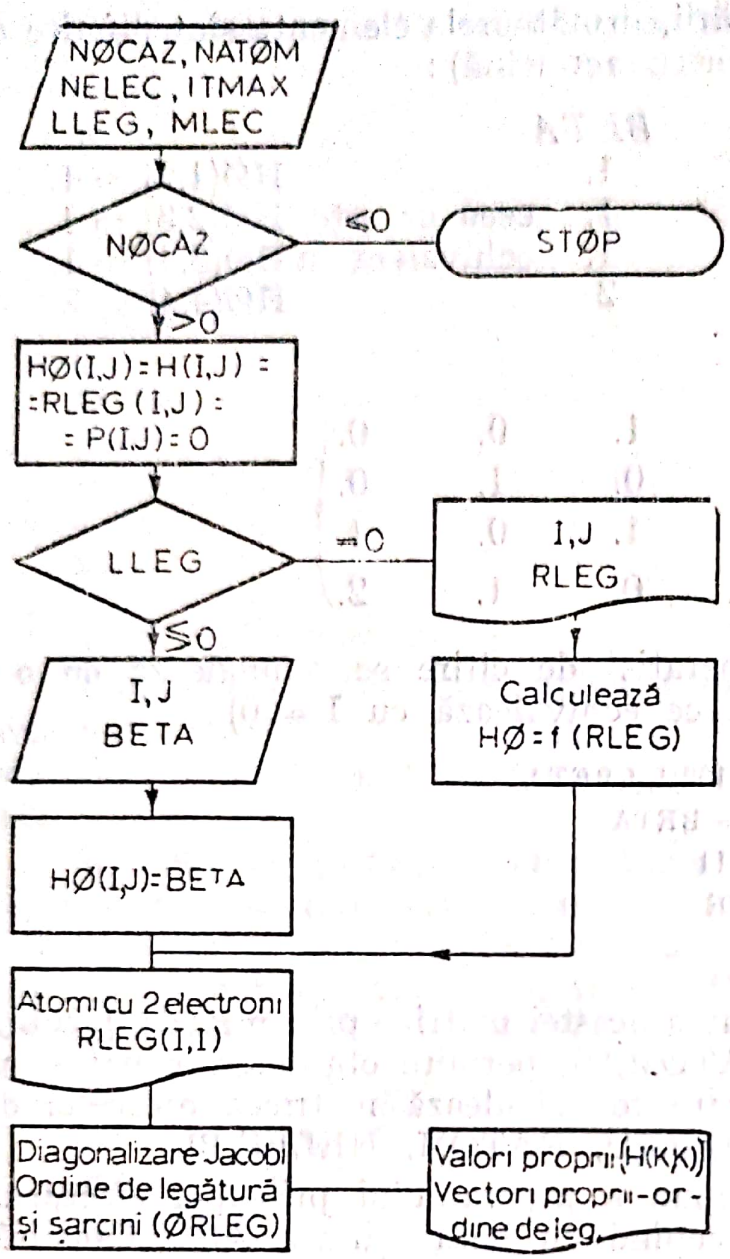


Fig. 3.2

un nou ciclu de iterare prin transfer la instrucţiunea 3000 (GØ TØ 3000).

NGATA este un indice cu ajutorul căruia se face tipărirea datelor finale. Acest indice a fost introdus pentru a se oferi posibilitatea tipăririi matricei Hückel în forma finală. Criteriul de convergenţă, după cum s-a văzut, este cel energetic. Calculul energiei necesită însă diagonalizarea matricei Hückel. În această operaţie, pentru a economisi memorie, se obişnuieşte ca valorile proprii să se transfere de la subrutina de diagonalizare, la programul principal, cu ajutorul aceleiaşi matrici, *H*, ceea ce înseamnă că matricea *H* iniţială este distrusă. Totuşi, uneori este necesar să cunoaştem şi forma finală a matricii *H*,

Scanned with OKEN Scanner

pentru a permite analiza modului în care elementele ei s-au modificat în raport cu valorile lor inițiale. Din acest motiv, a fost introdus indexul NGATA = 1, ca odată ce s-a realizat convergența să se reia calculul, pentru refacerea matricei H, în forma ei finală, pentru a fi tipărită pe imprimantă. Urmează operațiile obișnuite pentru calculul valorilor proprii, vectorilor proprii și a matricei de densitate (ordine de legătură și sarcini).

Organigrama părții iterative a programului principal este următoarea :

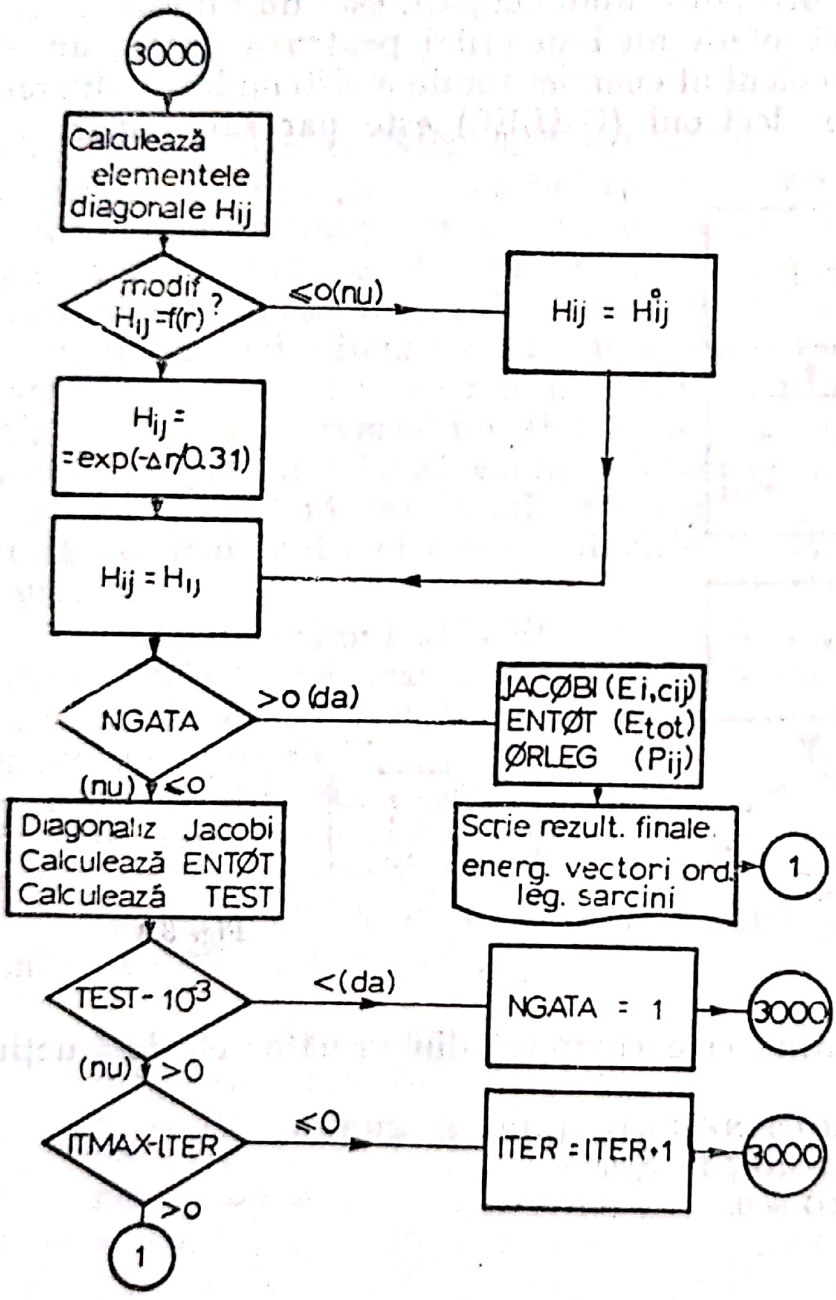


Fig. 3.3

Subrutinele programului ØHUC sînt:

1. Subprogramul ENTØT: servește la calculul energiei totale a sistemului electronic, prin însumarea valorilor energiilor corespunzătoare orbitalilor moleculari ocupați:

$$E_{tot} = \sum_{k=1}^{ocup} n_k \cdot E_k. \qquad (3.1.30)$$

Numărul de ocupare n_k este egal cu 1 sau 2, după cum în orbitalul respectiv se află unul sau doi electroni. Dacă molecula posedă orbitali simplu ocupați, este un radical liber. Această subrutină oferă un bun prilej pentru a ilustra un algoritm ingenios de calcul al energiei totale a sistemului, indiferent dacă numărul de electroni (NELEC) este par sau impar.

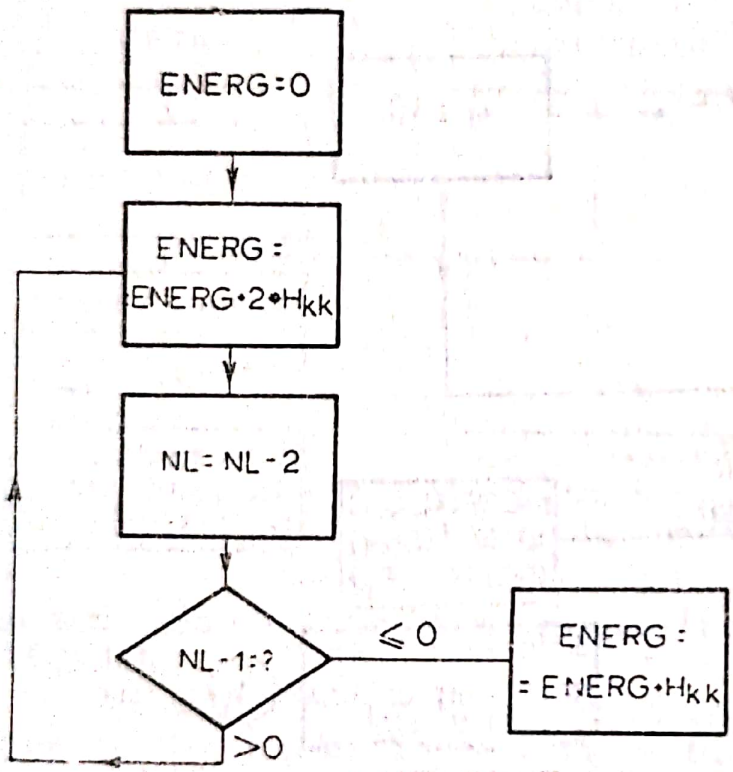


Fig. 3.4

Subrutina este compusă din următoarele instrucțiuni:

```
SUBRØUTINE ENTØT(NEL, H, ENERG)
DIMENSIØN H(20,20)
ENERG = 0.
NL = NEL
I = 0
```

```
3 I = I + 1
  ENERG = ENERG + 2.*H(I,I)
  NL = NL-2
  IF(NL-1)1,2,3
2 I = I + 1
  ENERG = ENERG + H(I,I)
1 CØNTINUE
  RETURN
  END
```

Secvenţa de chemare din programul principal este:

```
CALL ENTØT(NELEC, H, ENERG)
```

adică NL este egal cu NELEC(numărul de electroni). Deoarece la început I primeşte valoarea 1(prin instrucţiunea 3) însumarea energiilor începe cu nivelul energetic cel mai de jos. Pe măsură ce se înaintează spre nivelele superioare, numărul de electroni este micşorat din doi în doi, verificîndu-se mereu cîţi electroni au mai rămas: dacă nu a mai rămas nici unul valoarea obţinut pentru energie se transferă programului principal, dacă a mai rămas un singur electron energia ultimului orbital ocupat se adaugă energiei totale nemultiplicată cu doi şi apoi se transferă rezultatul programului principal, iar dacă au rămas mai mulţi electroni calculele se reiau de la instrucţiunea 3.

2. Subprogramul ØRLEG: are acelaşi principiu ca subrutina ENTØT. Se transferă din programul principal, prin secvenţa de chemare CALL ØRLEG(U, NATØM, NELEC, P), vectorii proprii U, numărul de atomi NATØM, şi numărul de electroni NELEC, calculîndu-se apoi matricea de densitate *P*, care este transmisă programului principal. Organigrama acestei subrutine este dată în fig. 3.5.

Instrucţiunile corespunzătoare acestei organigrame, în limbaj FØRTRAN, vor fi:

```
SUBRØUTINE ØRLEG(U,N,NEL,P)
DIMENSIØN P(20,20), U(20,20)
NL = NEL
DØ 1 I = 1,N
DØ 1 J = 1,N
K = 0
```

```
3 K = K + 1
  P(I,J) = 0.0
  P (I,J) = P(I,J) + 2.*U(I,K)*U(J,K)
  NL = NL - 2
  IF(NL - 1)1,2,3
2 K = K + 1
  P(I,J) = P(I,J) + U(I,K)*U(J,K)
1 CONTINUE
  RETURN
  END
```

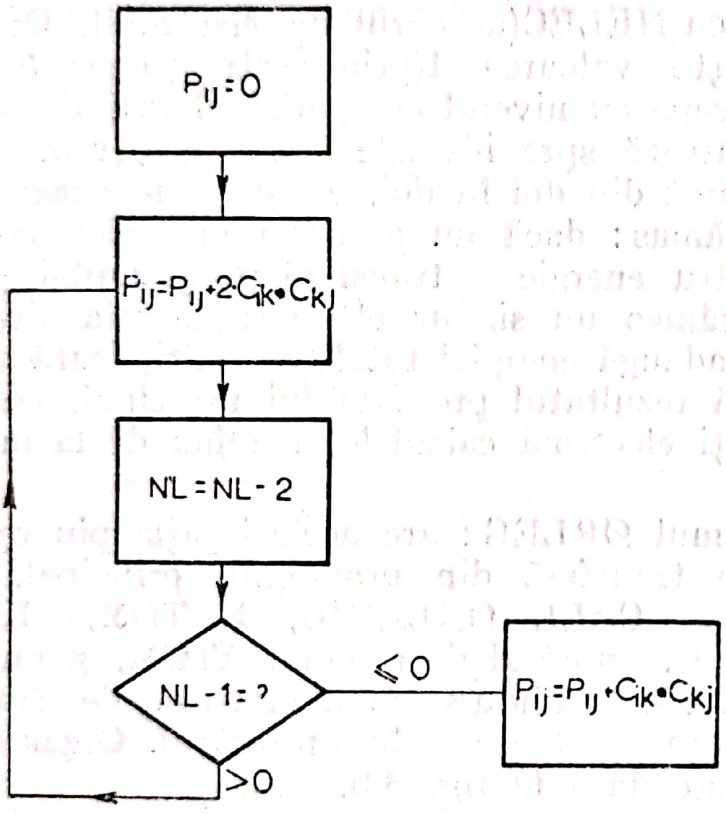


Fig. 3.5

3. Subrutinele JACØBI şi TIMAT au fost descrise la capitolul 2.4.

Programul principal va fi prezentat în rîndurile următoare. Comentariile care însoţesc programul sînt realizate cu scopul de a uşura urmărirea etapelor principale de calcul.

Scanned with OKEN Scanner

```
C
C     PRØGRAM DE CALCUL A ØRBITALILOR MØLECULARI PRIN METØDA HUECKEL,
C     TEHNICA ØMEGA (CU VALØAREA -1.4), PENTRU 20 DE CENTRE PI.
C     IN REPREZINTA NR. UNITATI DE CITIRE, IØ ESTE UNITATE DE SCRIERE A
C     CARØR VALØRI DEPIND DE CALCULATØRUL FØLØSIT. NU UITATI INITIALIZAR
C     EA LØR.
C
      DIMENSIØN H(20,20), HØ(20,20), RLEG(20,20), U(20,20), P(20,20), TITLU      (1ØHUK  01
     18)                                                                         ØHUK  02
C
C     INLØCUITI ACESTE CARTELE CU IN =                                          ØHUK  03
C                                  IØ =                                         ØHUK  04
C
    1 READ (IN, 100) NØCAZ, NATØM, NELEC, ITMAX, LLEG, MLEG                     ØHUK  05
      IF (NØCAZ) 1000,1000, 2000                                                ØHUK  06
C     NØCAZ ESTE NR. CAZULUI PE CARE IL STUDIATI, FØRMAT I2
C     DACA PRØBLEMELE AU FØST EPUIZATE FACETI NØCAZ = 0,
C     ADICA INTRØDUCETI LA SFIRSITUL SETULUI DE DATE Ø CARTELA
C     NEPERFØR ATA.
C
C     NATØM NR. ATØMILØR FACIND PARTE DIN SISTEMUL PI(I2)
C     NELEC NR. ELECTRØNILØR PI (I2)
C     ITMAX NR. MAXIM AL ITERATIILOR (I2)
C     LLEG = + 1 SE DAU LUNGIMILE LEGATURILØR PENTRU CALCULUL
C                INTEGRALELØR BETA (I2)
C     LLEG =   0 SE DAU VALØRILE BETA (I2)
C     MLEG = + 1 TERMENII NEDIAGØNALI SE CØNSIDERA DEPENDENTI
C                DE LUNGIMEA LEGĂTURILØR
 1000 STØP                                                                      ØHUK  07
```

Scanned with OKEN Scanner

```
2000 READ(IN,101) (TITLU(I), I = 1,18)                          ØHUK 08
     DØ 2 I = 1, NATØM                                          ØHUK 09
     DØ 2 J = 1, NATØM                                          ØHUK 10
     HØ (I,J) = 0.                                              ØHUK 11
     H(I,J) = 0.                                                ØHUK 12
     RLEG (I,J) = 0.                                            ØHUK 13
     P(I,J) = 0.                                                ØHUK 14
   2 CØNTINUE                                                   ØHUK 15
     IF (LLEG) 10,11,10                                         ØHUK 16
C
C    I SI J REPREZINTA CEI DØI ATØMI INTRE CARE EXISTA LEGATURA
C    AVIND INTEGRALA DE REZØNANTA BETA SAU Ø LEGATURA DE LUNGIME REGL.
C    FØRMAT(2I2,F10.5). DACA I = J, ATUNCI BETA REPREZINTA INTEGRALA
C    CØULØMBIANA A HETERØATØMULUI I, URMEAZA CARTELA GØALA.
  10 READ(IN,102) I,J, RLEG(I,J)                                ØHUK 17
     IF(I) 12,12,13                                             ØHUK 18
C
C    TERMENII NEDIAGØNALI PENTRU LEG.C—C SE CALCULEAZA CU
C    FØRMULA LØNGUET-HIGGINS SI SALEM. URMEAZA Ø CARTELA GØALA.
C
  13 HØ(I,J) = EXP (—(RLEG(I,J)—1.397)/.3106)                   ØHUK 19
     H(I,J) = HØ (I,J)                                          ØHUK 20
     GØ TØ 10
  12 WRITE (IØ, 1Ø3)                                            ØHUK 21
     CALL TIMAT (RLEG, NATØM, 0,IØ)                             ØHUK 22
     GØ TØ 16                                                   ØHUK 23
  12 READ (IN,102) I,J, BETA                                    ØHUK 24
     HØ (I,J) = BETA                                            ØHUK 25
     H(I,J) = BETA                                              ØHUK 26
```

Scanned with OKEN Scanner

```
      IF(I) 14,14,11                                              ØHUK  27
   14 CØNTINUE                                                    ØHUK  28
C
C     EXISTA HETERØATØM CØNTRIBUIND CU DØI ELECTRØNI
C     DACA RASPUNSUL ESTE AFIRMATIV, MENTIØNATI-L  AICI.
C
   16 READ(IN, 102) I,I, RLEG (I,I)                               ØHUK  29
      IF(I) 15,15,16                                              ØHUK  30
   15 CØNTINUE                                                    ØHUK  31
      DØ 17 I = 1, NATØM                                          ØHUK  32
      IF(RLEG (I,I)) 17,17,18                                     ØHUK  33
   18 WRITE(IØ,104) I, RLEG(I,I)                                  ØHUK  34
   17 CØNTINUE                                                    ØHUK  35
      WRITE (IØ,105)                                              ØHUK  36
      CALL TIMAT (HØ, NATØM, 0, IØ)                               ØHUK  37
C
C     IN FINE, INCEP ITERATIILE.
C
      ITER = 0                                                    ØHUK  38
      ENANT = 0.                                                  ØHUK  39
C
C     LA INCEPUT SE CALCULEAZA VALØRILE PRØPRII
C     ŞI VECTØRII PRØPRII.
C
      CALL JACØBI(HØ, NATØM,U)                                    ØHUK  40
      WRITE(IØ,106)                                               ØHUK  41
      WRITE(IØ,107) (HØ(K,K), K = 1, NATØM)                       ØHUK  42
      WRITE(IØ,108)                                               ØHUK  43
      CALL TIMAT (U,NATØM, 0, IØ)                                 ØHUK  44
```

Scanned with OKEN Scanner

```
C
C     PE URMA MATRICEA DE DENSITATE.
C
      CALL ØRLEG (U,NATØM, NELEC, P)                          ØHUK  45
      WRITE (IØ,109)                                          ØHUK  46
      CALL TIMAT (P,NATØM, 0, IØ)                             ØHUK  47
C
C     RECALCULAREA TERMENILØR DIAGØNALI
C
 3000 DØ 21 I = 1, NATØM                                      ØHUK  48
      IF (RLEG(I,I)-1.) 22,22,23                              ØHUK  49
   22 H(I,I) = HØ(I,I) + 1.4*(1.-P (I,I))                     ØHUK  50
      GØ TØ 21                                                ØHUK  51
   23 H(I,I) = HØ(I,I) + 1.4*(2.-P(I,I))                      ØHUK  52
   21 CØNTINUE                                                ØHUK  53
      NGATA = 0                                               ØHUK  54
      IF (MLEG) 23,25,23                                      ØHUK  55
   23 M = NATØM-1                                             ØHUK  56
C
C     CE FACEM CU TERMENUL NEDIAGØNAL? DECIDE VALØAREA VARI-
C     ABILEI MLEG. DAR, ØARE CUM AM ALES-Ø? VEZI PRIMA CARTELA CU
C     DATE! DACA H(I,J) ESTE FUNCŢIE DE LLEG (I,J) ATUNCI H(I,J) ESTE
C     EGAL CU FØRMULA LØNGUET-HIGGINS ŞI SALEM; DACA MLEG = 0,
C     H(I,J) = HØ (I,J), ADICA MATRICEA HUECKEL INIŢIALA.
C
      DØ 24 I = 1,M                                           ØHUK  57
      DØ 24 J = I + 1, NATØM                                  ØHUK  58
      LLEG(I,J) = 1.517-Ø.18*P(I,J)                           ØHUK  59
      H(I,J) = EXP(-(RLEG(I,J)-1.397)/.3106)                  ØHUK  60
```

Scanned with OKEN Scanner

```
   24 CØNTINUE                                                ØHUK  61
      GØ TØ 27                                                ØHUK  62
   25 M = NATØM-1                                             ØHUK  63
      DØ 26 I = 1,M                                           ØHUK  64
      DØ 26 J = I + 1, NATØM                                  ØHUK  65
      H(I,J) = HØ(I,J)                                        ØHUK  66
   26 CØNTINUE                                                ØHUK  67
   27 DØ 28 I = 1, NATØM                                      ØHUK  68
      DØ 28 J = 1, NATØM                                      ØHUK  69
      H(J,I) = H(I,J)                                         ØHUK  70
   28 CØNTINUE                                                ØHUK  71
      IF(NGATA) 34,34,33                                      ØHUK  72
   34 CALL JACØBI(H,NATØM,U)                                  ØHUK  73
      CALL ENTØT(NELEC, H, ENERG)                             ØHUK  74
      WRITE(IØ,110) ITER                                      ØHUK  75
      TEST = ABS(ENANT-ENERG)                                 ØHUK  76
C
C     TESTUL DE CØNVERGENTA ENERGETICA
C
      IF(TEST-0.001) 29,29,30                                 ØHUK  77
   29 NGATA = 1                                               ØHUK  78
      WRITE(IØ,111) ITER                                      ØHUK  79
      GØ TØ 3000                                              ØHUK  80
C
C     CØNFIRMAREA REUSITEI. SUCCES LA CAZUL URMATØR.
C
   33 WRITE (IØ,113)                                          ØHUK  81
      WRITE(IØ,114)                                           ØHUK  82
      CALL TIMAT (H,NATØM,0,IØ)                               ØHUK  83
```

Scanned with OKEN Scanner

```
      CALL JACØBI(H,NATØM,U)                                          ØHUK  84
      WRITE(IØ,115)                                                   ØHUK  85
      WRITE(IØ,107) (H(K,K), K = 1, NATØM)                            ØHUK  86
      WRITE(IØ,116)                                                   ØHUK  87
      CALL TIMAT(U,NATØM,O,IØ)                                        ØHUK  88
      WRITE (IØ,117)                                                  ØHUK  89
      CALL ØRLEG(U,NATØM,  NELEC,P)                                   ØHUK  90
      CALL TIMAT(P,NATØM, O, IØ)                                      ØHUK  91
      GØ TØ 1                                                         ØHUK  92
C
C     NU CUMVA AM DEPASIT NR. ITERATIILØR?
C
   30 IF (ITMAX-ITER) 31,31,32                                        ØHUK  93
   31 WRITE (IØ,112) ITER                                             ØHUK  94
      GØ TØ 1                                                         ØHUK  95
   32 ITER = ITER + 1                                                 ØHUK  96
      GØ TØ 3000                                                      ØHUK  97
  100 FØRMAT(6I2)                                                     ØHUK  98
  101 FØRMAT(18A4)                                                    ØHUK  99
  102 FØRMAT(2I2,F10.5)                                               ØHUK 100
  103 FØRMAT(///,'MATRICEA LUNGIMII LEGATURILØR ESTE',///)            ØHUK 101
  104 FØRMAT(12HHETERØATOMUL,I2,12HCØNTRIBUE CU,I2,9HELECTØNI)        ØHUK 102
  105 FØRMAT(///,'MATRICEA HUECKEL INIŢIALA',///)                     ØHUK 103
  106 FØRMAT(///,'NIVELELE ENERGETICE INITIALE SINT',///)             ØHUK 104
  107 FØRMAT(30X,F10.5)                                               ØHUK 105
  108 FØRMAT(///,'FUNCTIILE DE UNDA INIŢIALE SINT',///)               ØHUK 106
  109 FØRMAT(///,'MATRICEA ØRDINELØR DE LEGATURA INIŢIALE ESTE',///)  ØHUK 107
  110 FØRMAT(10X,5HDUPA, I3,39HITERAŢII VALØAREA ENERGIEI TØTALE      ØHUK 108
     1ESTE, F 10.5)                                                   ØHUK 109
```

Scanned with OKEN Scanner

```
111 FØRMAT(///,30HENERGIA ESTE CØNSTANTA DUPA,  I3,8HITERATII)      ØHUK 110
112 FØRMAT(//,31HENERGIA NU ESTE CØNSTANTA DUPA, I3,8HITERAȚII)     ØHUK 111
113 FØRMAT(//,' VALØRILE FINALE SINT URMATØARELE',//)               ØHUK 112
114 FØRMAT ('MATRICEA HUECKEL',///)                                 ØHUK 113
115 FØRMAT(10X,'NIVELELE ENERGETICE SINT')                          ØHUK 114
116 FØRMAT(///,' FUNCȚIILE DE UNDA',///)                            ØHUK 115
117 FØRMAT('MATRICEA ØRDINELOR DE LEGATURA SI SARCINII ELEC'
   1'TRØNICE')                                                      ØHUK 116
    END                                                             ØHUK 117
```

Scanned with OKEN Scanner

3.1.4 CALCULUL STRUCTURII ELECTRONICE ÎN APROXIMAŢIA PARISER—PARR—POPLE

Apariţia calculatoarelor comerciale cu memorie internă mare a impulsionat programarea unor teorii cu grade de complexitate mult superioare metodei Hückel. Aşa cum am menţionat în capitolul introductiv, ecuaţiile fundamentale ale sistemelor moleculare nu se pot aplica, fără introducerea unor aproximaţii. Un moment de reviriment, din punct de vedere practic, a fost marcat odată cu apariţia lucrărilor lui Pariser şi Parr şi independent Pople[15], care, păstrînd numeroase valenţe riguroase ale ecuaţiilor Roothaan, au introdus o parametrizare, cunoscută astăzi sub numele de PPP (de la iniţialele autorilor) şi care aplicată la foarte numeroase cazuri a oferit satisfacţie. Din acest motiv, deşi amploarea calculelor de tip PPP este astăzi mult redusă, totuşi această metodă continuă să fie utilizată atunci cînd obiectul de interes îl reprezintă un sistem π.

Calculul de tip PPP este autocoerent, deci necesită reluarea operaţiilor în mai multe cicluri pînă cînd se satisface un criteriu de convergenţă (v.pag. 133). Maratonul calculelor numerice cuprinde (i) rezolvarea unui număr substanţial de integrale, cu ajutorul unor formule destul de complicate, (ii) calculele care vizează elementele de matrice ale hamiltonianului sînt reluate pînă cînd se realizează un criteriu de convergenţă, la fiecare ciclu executîndu-se şi o diagonalizare. Toate acestea reprezintă numai cîteva aspecte care explică de ce era calculelor cu metode elaborate este legată de dezvoltarea calculatoarelor.

Programul care va fi descris în rîndurile care urmează se bazează pe ecuaţiile standard, folosite în mod curent, în calculele de tip Pariser-Parr-Pople. Matricea Fock, F, se calculează conform următoarelor relaţii:

$$F(i,i) = W_i + \frac{1}{2} \times q_i(ii,ii) + \sum_{j \neq i} (q_j - 1)\ (ii,jj) \qquad (3.1.31)$$

$$F(i,j) = \beta_{ij} - \frac{1}{2} p_{ij} \times (ii,jj), \qquad (3.1.32)$$

unde:

W_i — potenţial de ionizare

q_i şi q_j — sarcina electronică a atomilor „i" şi „j",

Scanned with OKEN Scanner

$$(ii,\ ii) = \int \Phi_i(1) \cdot \Phi_i(1) \frac{1}{r_{12}} \Phi_i(2) \cdot \Phi_i(2)\ dV$$

$$(ii,\ jj) = \int \Phi_i(1) \cdot \Phi_i(1) \frac{1}{r_{12}} \Phi_j(2) \cdot \Phi_j(2)\ dV$$

și

β_{ij} — integrala de rezonanță

p_{ij} — ordinul de legătură între centrul „i" și „j" (v. def. la pag. 133). În program, sarcinile electronice și ordinele de legătură alcătuiesc o singură matrice P, avînd ca elemente diagonale q_i, în timp ce elementele nediagonale sînt p_{ij}.

Atît W_i din formula (3.1.31), cît și β_{ij} din formula (3.1.32) se tratează ca parametri semiempirici, valorile fiind alese în urma unor calcule de calibrare efectuate pe cazuri simple, în așa fel încît să se reproducă anumite constante fizice (lungimi de legătură, unghiul de valență, moment de dipol etc.). Tabelul 3.2 și 3.3 prezintă pentru cîțiva atomi aparținînd perioadei a doua, valori recomandate pentru W_i și β_{ij} (cititorul, în funcție de date de literatură sau opțiuni personale, poate folosi și alte valori). Se tratează de asemenea semiempiric și integrala de repulsie monocentrică, (ii,ii); valoarea ei se alege ca fiind egală

TABELUL 3.2.

VALORI RECOMANDATE PENTRU POTENȚIALE DE IONIZARE W_i, INTEGRALE DE REPULSIE MONOCENTRICE $(ii,\ ii)$ ȘI EXPONENTE SLATER PENTRU UNII ATOMI DIN PERIOADA A II-a

Atom	W_i	$(ii,\ ii)$	Exponent Slater
C	11.16	10.84	3.25
N	14.12	12.43	3.95
N̈	25.73	16.76	3.90
O	17.70	17.70	4.90

TABELUL 3.3.

VALORI ALE INTEGRALEI DE REZONANȚĂ, β_{ij}, PENTRU O PARTE DIN ATOMII PERIOADEI A II-a

	C	N̈	Ö
C	2.30	2.45	2.90
N	2.45	2.35	—
O	2.90	—	—

Scanned with OKEN Scanner

cu diferența dintre potențialul de ionizare a electronului, aflat în starea de valență corespunzătoare (P_I) și afinitatea electronului (P_A):

$$(ii, ii) = P_I - P_A \tag{3.1.33}$$

Uneori, se preferă folosirea unor relații funcționale pentru calculul integralei β_{ij}. Foarte mult utilizată este o dependență funcțională de forma:

$$\beta_{ij} = a \cdot e^{-br_{ij}} \text{ (în eV)}, \tag{3.1.34}$$

unde a și b sînt constante, pentru care, în tabelul 3.4 se dau cîteva valori, iar r_{ij} este distanța între centrele atomice „i" și „j".

TABELUL 3.4.

a	b	lit.
— 2.517,5	5,007	16
— 167,9	3,023	17
—61.232,0	2,3181	18

Un alt tip de aproximații, folosit în mod deosebit în anii din urmă, este relația lui Wolfsberg și Helmholtz[19]

$$\beta_{ij} = \frac{1}{2} S_{ij}(W_i + W_j), \tag{3.1.35}$$

unde W_i și W_j sînt potențiale de ionizare (v. tabelul 3.2).

Există, de asemenea, mai multe procedee de evaluare și a integralei bicentrice (ii,jj). Nu este în intenția noastră de a discuta în mod critic aceste aproximații, cititorul poate insera în programul său PPP acea aproximație pe care o consideră că se potrivește cel mai bine scopului ales.

a) aproximația Parr[20]

— pentru cazul homonuclear:

$$(ii, jj) = Z \cdot \frac{e^2}{a_0} \times [(1/\rho) - (12/\rho^3) + (324/\rho^5)], \tag{3.1.36}$$

unde Z este exponentul Slater (v. tabelul 3.2),

a_0 — prima rază Bohr, 0.5292 A,

$\rho = ZR/a_0$, R fiind distanța între cei doi atomi,

Scanned with OKEN Scanner

— pentru cazul heteronuclear:

$$(ii, jj) = e^2\{R^{-1} - [(q_A + q_B)/4R^3] + (9 \cdot q_A \cdot q_B/16R^5)\} \quad (3.1.37)$$

unde

$q_A = 24 \cdot a_0^2/Z_A^2$ şi $q_B = 24\ a_0/Z_B^2$ cu semnificaţiile variabilelor identice cu cele din ecuaţia (3.1.36).

b) Nishimoto Mataga[21]

$$(ii, jj) = 14.3897/(a_{ij} + R_{ij}) \text{ (în eV)} \quad (3.1.38)$$

în care:

$$a_{ij} = 2{\cdot}0 \times 14{\cdot}3897/(1/(ii, ii) + 1/(jj, jj) \quad (3.1.39)$$

c) Aproximaţia Ohno[22]

$$(ii, jj) = \frac{e^2}{R_{ij}^2 + a_{ij}}, \quad (3.1.40)$$

unde a_{ij} se calculează cu formula (3.1.39).

d) Unii autori preferă calculul integralelor de repulsie prin două procedee. Dacă $R \leqslant 2{,}8$ Å se preferă una din aproximaţiile menţionate, adică a — c; pentru $R > 2{,}8$Å, integralele de repulsie se calculează teoretic. În cazul în care se utilizează orbitali Slater, integralele de repulsie se calculează cu formulele lui Roothaan[23]. Aceste formule sînt relativ complicate. Avînd în vedere că în literatura de specialitate românească sînt destul de greu accesibile, vrem să venim în ajutorul cititorului şi să prezentăm, în rîndurile care urmează, numai o parte din ele, potrivit cazurilor discutate în acest subcapitol. Din punctul de vedere al programării, această ocazie ne va permite să ilustrăm utilizarea funcţiilor externe (v.1.4.1). Aparent, aceste formule par a fi foarte complicate. Notaţiile suplimentare au fost introduse cu scopul de a facilita recunoaşterea termenilor respectivi, în codificarea FØRTRAN.

ζ este exponentul Slater, iar R corespunde distanţei dintre centrele a şi b; a şi b sînt noii indici, care substituie pe i şi j. Această substituţie nu reprezintă o complicaţie suplimentară datorită unei neatenţii sau inconsecvenţe, ci o facilitare în codificarea variabilelor care conţin aceşti indici.

Scanned with OKEN Scanner

Se mai introduc notaţiile (în paranteză denumirea în program)

$$\zeta = \frac{1}{2} \times (\zeta_a + \zeta_b) \quad \text{(ZETA)} \tag{3.1.41}$$

$$\tau = (\zeta_a - \zeta_b)/(\zeta_a + \zeta_b) \quad \text{(TAU)}$$

$$\rho = \frac{1}{2}(\zeta_a + \zeta_b) \cdot R \quad \text{(RHO)}$$

$$K = \frac{1}{2}(\tau + 1/\tau) \quad \text{(KAP)}$$

$$\rho_a = \zeta_b \cdot R \quad \text{(RHOA)}$$

$$\rho_b = \zeta_b \cdot R \quad \text{(RHOB)}$$

$$\tau_a = (\zeta_a - \zeta_a^{-1})/(\zeta_a + \zeta_a^{-1}); \; \tau_b = (\zeta_b - \zeta_b^{-1})/(\zeta_b + \zeta_b^{-1}).$$

Modificăm notaţia (ii,jj) în (aa,bb), în scopul de a fi fideli formulelor originale din literatură:

$$(ii,\ jj) = (aa,\ bb) = (1+\tau_a)^{5/2} \cdot (1-\tau_a)^{5/2} \cdot (1+\tau_b)^{5/2} \cdot (1-\tau_b)^{5/2}$$

$$[3S_a|3S_b] - 3/2 \cdot [3S_a|3D\,\Sigma_b] - 3/2 \cdot [3D\,\Sigma_a|3S_b] + 9/4 \cdot [3D\Sigma_a|3D\Sigma_b] + 27/4 \cdot [3D\Delta_a|3D\Delta_b] \tag{3.1.42}$$

Termenii din parantezele mari sînt integrale de acoperire cu semnificaţii care se vor prezenta în cele ce urmează. Pentru a facilita urmărirea modului de programare, vom prezenta simultan cu notaţiile din formulele (3.1.42) şi prescurtările variabilelor respective utilizate în program

$$\text{ACOSS} = [3S_a|3S_b] = (1/R)\ [1 - (1-k)^3 \{\text{T1A} + \text{T2A} \cdot \rho_a + \text{T3A} \cdot \rho_a^2 - 1/24 \cdot k \cdot \rho_a^3\} e^{-2\rho_a} - (1+k)^3 \{\text{T1B} + \text{T2B} \cdot \rho_b + \text{T3B} \cdot \rho_b^2 + 1/24 \cdot k \cdot \rho_b^3\} e^{-2\rho_b}] \tag{3.1.43}$$

Scanned with OKEN Scanner

unde, pentru simplificare, s-au definit

$$
\begin{aligned}
T1A &= (1/32)(21 + 44k + 35k^2 + 10k^3)(1 + 2\rho_a) \\
T2A &= (1/24)(29 + 56k + 40k^2 + 10k^3)\rho_a^2 \\
T3A &= (1/12)(8 + 12k + 5k^2)\rho_a^3 \\
T4A &= (1/24)(5 + 4k)\rho_a^4 \\
T5A &= (1/36)\rho_a^5 \\
T1B &= (1/32)(11+7k-39k^2+35k^3-10k^4)(1+2\rho_b) \\
T2B &= (1/24)(14 + 13k - 51k^2 + 40k^3 - 10k^4)\rho_b^2 \\
T3B &= (1/12)(3 + 6k - 12k^2 + 5k^3)\rho_b^3 \\
T4B &= (1/24)(1 + 5k - 4k^2)\rho_b^4 \\
T5B &= (1/36)k\rho_b^5 .
\end{aligned}
\tag{3.1.46}
$$

Funcţia-subprogram care efectuează aceste calcule va conţine următoarele instrucţiuni:

```
      FUNCTIØN ACØSD (ZETA, TAU, RHØ, KAP, RHØA, RHØB)
      REAL KAP, KAP2, KAP3, KAP4
      KAP2 = KAP**2
      KAP3 = KAP**3
      KAP4 = KAP**4
      T1A = (1./32.)*(21. + 44.*KAP + 35.*KAP2 + 10.*KAP3)*
     1      (1. + 2.*RHØA)
      T2A = (1./24.)*(29. + 56.*KAP + 40.*KAP2 + 10.*KAP3)*
     1      (RHØA**2)
      T3A = (1./12.)*(8. + 12.*KAP + 5.*KAP2)  *(RHØA**3)
      T4A = (1./24.)*(5. + 4.*KAP)*(RHØA**4)
      T5A = (1./36.)*(RHØA**5)
      T1B = (1./32.)*(11. + 7.*KAP-39.*KAP2 + 35.*KAP3-10.
     1      *KAP4)*(1. + 2.*RHØB)
      T2B=(1./24.)*(14.+13.*KAP-51.*KAP2+40.*KAP3-10.
     1      *KAP4)*(RHØB**2)
      T3B = (1./12.)*(3. + 6.*KAP-12.*KAP2 + 5.*KAP3)*
     1      (RHØB**3)
      T4B = (1./24.)*(1. + 5.*KAP-4.*KAP2)*(RHØB**4)
      T5B = (1./36.)*KAP*(RHØB**5)
      ACØSD = (ZETA/(((1.-TAU)**2)*(RHØ**3)))*(1.-
     1      ((1.-KAP)**4)*(T1A + T2A + T3A + T4A + T5A) * EXP
     2      (-2.*RHØA)- ((1. + KAP)**3)* (T1B + T2B + T3B + T4B
     3      +T5B)*EXP(-2.*RHØB)))
      RETURN
      END
```


Scanned with OKEN Scanner

unde:

$$T1A = \frac{1}{16}(8 - k - 27k^2 - 30k^3 - 10k^4)$$

$$T1B = \frac{1}{16}(8 + k - 27k^2 + 30k^3 - 10k^4)$$

$$T2A = \frac{1}{32}(11 - 19k - 44k^2 - 20k^3) \qquad (3.1.44)$$

$$T2B = \frac{1}{32}(11 + 19k - 44k^2 + 20k^3)$$

$$T3A = \frac{1}{16}(1 - 5k - 4k^2)$$

$$T3B = \frac{1}{16}(1 + 5k - 4k^2)$$

$[3S_a|3S_b]$ denumită ACØSS (*ACØperire tip S*) este calculată de o funcţie-subprogram cu acelaşi nume. Deoarece KAP (prescurtarea lui k) ar fi o variabilă de tip întreg (prima literă este K), ea trebuie declarată reală, adică în program trebuie să apară instrucţiunea REAL KAP. Acest procedeu va fi întrebuinţat la toate subrutinele şi funcţiile descrise, ori de cîte ori se va utiliza variabila KAP.

```
      FUNCTIØN ACØSS(ZETA, TAU, RHØ, KAP, RHØA, RHØB)
      REAL KAP
      T1A = (1./16.)*(8.- KAP-27.*KAP**2-30.*KAP**3-
     1      10.*KAP**4)
      T1B = (1./16.)*(8. + KAP-27.*KAP**2 + 30.*KAP**3-
     1      10.*KAP**4)
      T2A = (1./32.)*(11.-19.*KAP-44.*KAP**2-20.*KAP**3)
      T2B = (1./32.)*(11. + 19.*KAP-44.*KAP**2 + 20.*KAP**3)
      T3A = (1./16.)*(1.-5.*KAP-4.*KAP**2)
      T3B = (1./16.)*(1.+5.*KAP-4.*KAP**2)
      ACØSS = (ZETA/RHØ)*(1.-((1.-KAP)**3)*(T1A + RHØA*T2A
     1      +(RHØA**2)*T3A-(1./24.)*KAP*(RHØA**3))
     2      *EXP(-2.*RHØA))-((1. + KAP)**3)*(T1B + RHØB*T2B
     3      + (RHØB**2)*T3B + (1./24.)*KAP*(RHØB**3))
     4      *EXP(-2.*RHØB))
      RETURN
      END
```


Scanned with OKEN Scanner

Termenul $[3S_a|3D\Sigma_b]$, denumit ACØSD (*AC*Øperire *SD*) se calculează cu ajutorul formulei:

$$\text{ACØSD} = [3S_a|3D\Sigma_b] = (\rho/(1-\tau)^2\rho^3)(1-(1-k)^4(\text{T1A} + \text{T2A} + \text{T3A} + \text{T4A} + \text{T5A}) \cdot e^{-2\rho_a} - (1+k)^3(\text{T1B} + \text{T2B} + \text{T3B} + \text{T4B} + \text{T5B}) \cdot e^{-2\rho_b}) \quad (3.1.45)$$

Al treilea termen $[3D\Sigma_a|3S\Sigma_b]$ = ACØDS (*AC*operirea Ørbitalilor *D* şi *S*) se calculează din (3.1.45) dacă ρ_a se înlocuieşte cu ρ_b, k cu $-k$ şi τ cu $-\tau$.

Termenul $[3D\Sigma_a|3D\Sigma_b]$ = AC3DS (*AC*operirea orbitalilor de tip 3D cu 3S) rezultă din:

$$\text{AC3DS} = (6\zeta/(1+\tau)^2(1-\tau)^2\rho^5)(1-(1-k)^4 (\text{T1A} + \text{T2A} + \text{T3A} + \text{T4A} + \text{T5A} + \text{T6A} + \text{T7A})e^{-2\rho_a} - (1+k)^4(\text{T1B} + \text{T2B} + \text{T3B} + \text{T4B} + \text{T5B} + \text{T6B} + \text{T7B})e^{-2\rho_b}) \quad (3.1.47)$$

În expresia de mai sus s-au făcut următoarele prescurtări:

$$\begin{aligned}
\text{T1A} &= (1/32)(16 + 29k + 20k^2 + 5k^3)(1 + 2\rho_a) \\
\text{T2A} &= (1/144)(139 + 246k + 156k^2 + 40k^3)\rho_a^2 \\
\text{T3A} &= ((1/72)(43 + 72k + 45k^2 + 10k^3)\rho_a^3 \\
\text{T4A} &= (1/216)(57 + 88k + 50k^2 + 10k^3)\rho_a^4 \\
\text{T5A} &= (1/432)(39 + 52k + 20k^2)\rho_a^5 \\
\text{T6A} &= (1/216)(5 + 4k)\rho_a^6 \\
\text{T7A} &= (1/324)\rho_a^7 \\
\text{T1B} &= (1/32)(16 - 29k + 20k^2 - 5k^3)(1 + 2\rho_b) \\
\text{T2B} &= (1/144)(139 - 246k + 165k^2 - 40k^3)\rho_b^2 \\
\text{T3B} &= (1/72)(43 - 72k + 45k^2 - 10k^3)\rho_b^3 \\
\text{T4B} &= (1/216)(57 - 88k + 50k^2 - 10k^3)\rho_b^4 \\
\text{T5B} &= (1/432)(39 - 52k + 20k^2)\rho_b^5 \\
\text{T6B} &= (1/21)(5 - 4k)\rho_b^6 \\
\text{T7B} &= (1/324)\rho_b^7
\end{aligned} \quad (3.1.48)$$

Scanned with OKEN Scanner

Programul corespunzător acestor formule este:

```
      FUNCTIØN AC3DS(ZETA, TAU, RHØ, RHØA, RHØB)
      REAL KAP, KAP2, KAP3
      KAP2 = KAP**2
      KAP3 = KAP**3
      TIA = (1./32.)*(16. + 29.*KAP + 20.*KAP2 + 5.*KAP3)*
     1      (1. + 2.*RHØA)
      T2A = (1./144.)*(139. + 246.*KAP + 165.*KAP2 + 40.
     1      *KAP3)*RHØA**2
      T3A = (1./72.)*(43. + 72.*KAP + 45.*KAP2 + 10.*KAP3)
     1      *RHØA**3
      T4A = (1./216.)*(57. + 88.*KAP + 50.*KAP2 + 10.*KAP3)
     1      *RHØA**4
      T5A = (1./432.)*(39. + 52.*KAP + 20.*KAP2)*RHØA**5
      T6A = (1./216.)*(5. + 4.*KAP)*RHØA**6
      T7A = (1./324.)*(RHØA**7)
      T1B = (1./32.)*(16.-29.*KAP + 20.*KAP2-5*KAP3)*(1. + 2.*RHØB)
      T2B = (1./144.)*(139.-246.*KAP+165.*KAP2.-40.*KAP3)*RHØB**2
      T3B = (1./72.)*(43.-72.*KAP + 45.*KAP2-10.*KAP3) *RHØB**3
      T4B = (1./216.)*(57.-88.*KAP + 50.*KAP2-10.* KAP3)*RHØB**4
      T5B = (1./432.)*(39.-52.*KAP + 20.*KAP2)*RHØB**5
      T6B = (1./216.)*(5.-4.*KAP)*RHØB**6
      T7B = (1./324.)*RHØB**7
      AC3DS = (3.*ZETA/(((1. + TAU)**2)*((1.-TAU)**2)
     1      *(RHØ**5)))*(1.-((1.-KAP)**4)*(T1A + T2A +
     2      T3A + T4A + T5A + T6A + T7A)* EXP(-2.*RHØA)-((1.
     3      +KAP)**4)*(T1B + T2B + T3B + T4B + T5B + T6B + T7B)*
     4      EXP(-2.*RHØB)))
      RETURN
      END
```

În fine $[3D\Delta_a|3D\Delta_b]$ = AC3DD (*AC*operirea orbitalilor de tip 3*D* *D*elta) se calculează cu ajutorul formulei:

$$\begin{aligned}\text{AC3DD} = (\zeta/(1+\tau)^2(1-\tau)^2\rho^5)\ (1-(1-k)^4\,(\text{T1A} + \\ + \text{T2A} + \text{T3A} + \text{T4A} + \text{T5A})e^{-2\rho_a} \\ + (1+k)^4(\text{T1B} + \text{T2B} + \text{T3B} + \\ + \text{T4B} + \text{T5B})e^{-2\rho_b}) \qquad (3.1.49)\end{aligned}$$

Scanned with OKEN Scanner

Prescurtările folosite în această formulă sînt:

$$
\begin{aligned}
T1A &= (1/32)\ (16 + 29k + 20k^2 + 5k^3)\ (1 + 2\rho_a) \\
T2A &= (1/48)\ (43 + 72k + 45k^2 + 10k^3)\ \rho_a^2 \\
T3A &= (1/24)\ (11 + 14k + 5k^2)\ \rho_a^3 \\
T4A &= (1/24)\ (3 + 2k)\ \rho_a^4 \\
T5A &= (1/72)\ \rho_a^5 \\
T5A &= (1/32)\ (16 - 29k + 20k^2 - 5k^3)\ (1 + 2\rho_b) \\
T2B &= (1/48)\ (43 - 72k + 45k^2 - 10k^3)\ \rho_b^2 \\
T3B &= (1/24)\ (11 - 14k + 5k^2)\ \rho_b^3 \\
T4B &= (1/24)\ (3 - 2k)\ \rho_b^4 \\
T5B &= (1/72)\ \rho_b^5
\end{aligned}
\qquad (3.1.50)
$$

Subprogramul-funcţie corespunzător acestor formule este:

```
      FUNCTIØN AC3DD(ZETA, TAU, RHØ, KAP, RHØA, RHØB)
      REAL KAP, KAP2, KAP3
      KAP2 = KAP**2
      KAP3 = KAP**3
      T1A = (1./32.)*(16. + 29.*KAP + 20.*KAP2 + 5.*KAP3)*
     1      (1. + 2.*RHØA)
      T2A = (1./48.)*(43. + 72.*KAP + 45.*KAP2 + 10.*KAP3)
     1      *RHØA**2
      T3A = (1./24.)*(11. + 14.*KAP + 5.*KAP2)*RHØA**3
      T4A = (1./24.)*(3. + 2.*KAP)*(RHØA**4)
      T5A = (1./72.)*(RHØA**5)
      T1B = (1./32.)*(16.-29.*KAP + 20.*KAP2-5. *KAP3)
     1      *(1. + 2.*RHØB)
      T2B = (1./48.)*(43. - 72.*KAP + 45.*KAP2-10.*KAP3)
     1      *RHØB**2
      T3B = (1./24.)* (11.-14.*KAP + 5.*KAP2)*RHØB**3
      T4B = (1./24.)*  (3.-2.*KAP)*RHØB**4
      T5B = (1./72.)*RHØB**5
      AC3DD = (ZETA/(((1. + TAU)**2)*((1.-TAU)**2)*
     1        (RHØ**5)))* (1.-((1.-KAP)**4)*(T1A + T2A
     2        + T3A + T4A + T5A)*EXP (-2.*RHØA)) + (((1. + KAP)
     3        **4)*(T1B + T2B + T3B + T4B + T5B)*EXP (-2.*RHØB)))
      RETURN
      END
```

Aceste formule cu ajutorul cărora se evaluează integrale de tipul (*ii*,*jj*) într-o variantă teoretică conving cititorul de gradul complicaţiilor care intervin chiar dacă folosim metode aproximative de calcul.

Scanned with OKEN Scanner

De multe ori molecula conține și subsisteme homonucleare, de exemplu mai mulți atomi de carbon sau mai mulți atomi de azot. În astfel de cazuri, formulele prezentate înainte se simplifică foarte mult. Avînd în vedere frecvența cu care apar astfel de cazuri, este util ca în aceste situații să folosim următoarele formule:

$$[3S_a \mid 3S_b] = (\zeta/\rho)\ (1 - (1 + (419/256)\rho + (163/128)\rho^2 + (119/192)\rho^3 + (5/24)\rho^4 + (1/20)\rho^5 + (1/120)\rho^6 + (1/1200)\ \rho^7)\ e^{-2\rho} \qquad (3.1.51)$$

$$[3S_a \mid 3D\Sigma_b] = [3D\Sigma_a \mid 3S\Sigma_b] = (\zeta/\rho^3)\ (1 - (1 + 2\rho + 2\rho^2 + (4/3)\rho^3 + (2/3)\rho^4 + (191/120)\rho^5 + (31/360)\ \rho^6 + (19/840)\rho^7 + (17/3780)\rho^8 + (1/1890)\rho^9\ (e^{-2\rho}) \qquad (3.1.52)$$

$$[3D\Sigma_a/\ 3D\Sigma_b] = (6\zeta/\rho^5)(1 - (1 + 2\rho + 2\rho^2 + (4/3)\rho^3 + (2/3)\rho^4 + (511/1920)\rho^5 + (253/2880)\rho^6 + (2237/90720)\rho^7 + (71/11340)\rho^8 + (1/630)\rho^9 + (13/34020)\rho^{10} + (1/17050)\ \rho^{11})\ e^{-2\rho}) \qquad (3.1.53)$$

$$[3D\Delta_a \mid 3D\Delta_b] = (\zeta/\rho^5)\ (1 - (1 + 2\rho + 2\rho^2 + (4/3)\rho^3 + (2/3)\rho^4 + (253/960)\rho^5 + (119/1440)\rho^6 + (11/560)\rho^7 + (11/315)\rho^8 + (1/3780)\ \rho^9)\ e^{-2\rho}) \qquad (3.1.54)$$

Funcțiile-subprogram corespunzătoare acestor formule sînt:

```
      FUNCTIØN ACØSS(ZETA, RHØ)
      ACØSS = (ZETA/RHØ)*(1.-(1. + (419./256.)*RHØ +
     1        (163./128.)*(RHØ**2) + (119./192.)*(RHØ*
     2        *3) + (5./24.)*(RHØ**4) + (1./20.)*(RHØ**
     3        5) + (1./120.)*(RHØ**6) + (1./1200.)*(RHØ
     4        **7))*EXP(-2.*RHØ))
      RETURN
      END
      FUNCTIØN ACØSD( ZETA, RHØ)
      ACØSD =(ZETA/(RHØ**3))*(1.-(1. + 2.*RHØ + 2.* (RHØ
     1        **2) + (4./3.)*(RHØ**3) + (2./3.)* (RHØ**4)
     2        +(191./120.)*(RHØ**5) + (31./360.)* (RHØ**6) +
     3        (19./840.) *(RHØ**7) + (17./3780.)*(RHØ**8) +
     4        (1./1890.)*(RHØ**9))*EXP(-2.*RHØ))
      RETURN
      END
```


Scanned with OKEN Scanner

```
      FUNCTIØN AO3DS(ZETA, RHØ)
      AO3DS = (6.*ZETA/(RHØ**5))*(1.- (1. + 2.* (RHØ**2) +
     1        (4./3.)*(RHØ**3) + (2./3.)* (RHØ**4) + (511.
     2        /1920.)*(RHØ**5) + (253./2880.)*(RHØ**6) + (2237.
     3        /90720.)* (RHØ**7) + (71./11340.)*(RHØ**8) +
     4        (1./630.)*(RHØ**9) + (13./34020.)*(RHØ**10) +
     5        (1./17050.)*(RHØ**11))*EXP(-2.*RHØ))
      RETURN
      END

      FUNCTIØN AO3DD(ZETA, RHØ)
      AO3DD = (ZETA/(RHØ**5))*(1.-(1. + 2.*RHØ + 2.*(RHØ
     1        **2) + (4./3.)* (RHØ**3) + (2./3.)* (RHØ**4) + (253.
     2        /960.)* (RHØ**5) + (119./1440.) * (RHØ**6)
     3        + (11./560.)* (RØH**7) + (11./315.)* (RHØ**8)
     4        + (1./3780.)*(RHØ**9))*EXP(-2.*RHØ))
      RETURN
      END
```

Descrierea programului

Calculul elementelor de matrice (3.1.31) şi (3.1.32) necesită cunoaşterea coeficienţilor C_{ij} ai funcţiilor de undă moleculare. Procedeul de calcul şi în acest caz va fi identic cu acela folosit la metoda ØHUC, adică tehnica de calcul a coeficienţilor C_{ij} se bazează pe metoda cîmpului autocoerent.

În etapa iniţială, în mod evident $C_{ij} = 0$. Elementele matricei Fock devin:

$$F_{ii} = W_i,$$

unde

$$W_i = P_I - P_A, \tag{3.1.55}$$

şi

$$F_{ij} = \beta_{ij} \tag{3.1.56}$$

Prin diagonalizarea matricei F se obţine un set de vector proprii, în aproximaţia de ordinul zero, cu ajutorul cărora se calculează matricea ordinelor de legătură (sau matricea de densitate) cu care se calculează o nouă matrice Fock, F^1, dar de această dată utilizînd formulele (3.1.31) şi (3.1.32). Din diagonalizarea acestei matrice, va rezulta un nou set de vectori proprii, C_{ij}, utilizaţi la recalcularea matricii F. Calculul se repetă de atîtea ori, pînă cînd se verifică un criteriu de convergenţă. În programul care va fi prezentat, criteriul de convergenţă este

Scanned with OKEN Scanner

de natură energetică. În fiecare etapă a ciclului iterativ se va compara energia actuală, adică energia calculată în ultima iterație cu energia calculată în etapa anterioară. Se va compara fiecare valoare proprie. În cazul în care diferența $E_i - EANT_i$ este mai mică decît valoarea prescrisă, această condiție îndeplinindu-se pentru fiecare indice „i", calculul iterativ se oprește, deoarece există o autocoerență între două seturi consecutive de coeficienți C_{ij}.

În cazul în care criteriul de convergență nu se realizează, calculul iterativ este reluat de atîtea ori de cîte ori este nevoie pentru îndeplinirea criteriului de convergență. Există în practică cazuri în care convergența este fie prea lentă, fie nu se realizează

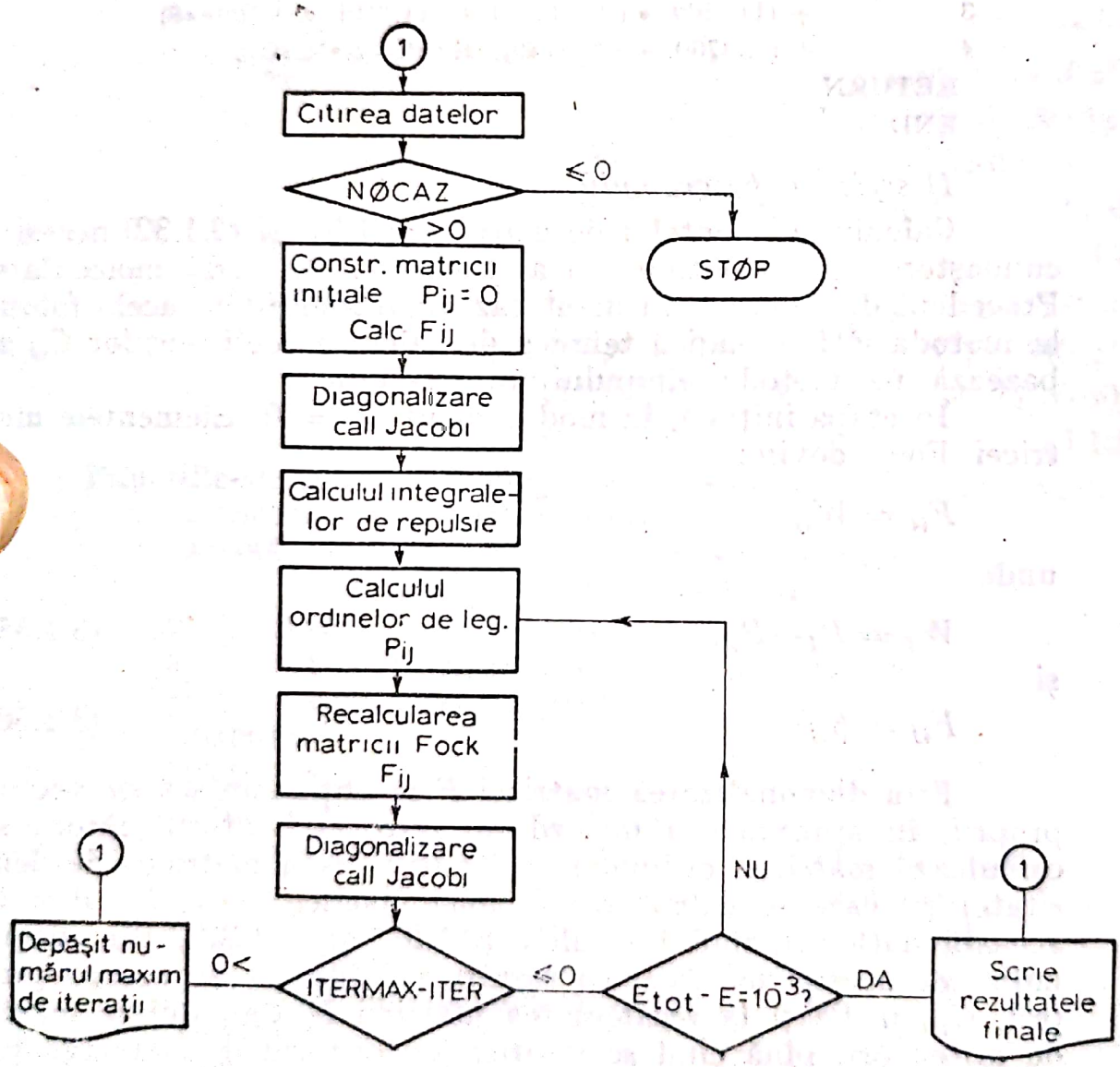


Fig. 3.6

niciodată (cazurile sistemelor divergente). Pentru a evita blocarea inutilă a calculatorului, cazurile sistemelor cu convergență lentă se tratează la fel ca și cazurile sistemelor divergente, calculul întrerupîndu-se după efectuarea unui anumit număr de iterații. În programul descris în paginile următoare, acest prag al iteraţiilor maxime este fixat la 20.

În ipoteza în care convergența a fost realizată, calculatorul va executa instrucțiuni de tipărire a unor date de interes, cum ar fi nivelele energetice π, matricea de densitate și funcțiile moleculare.

Dacă interesul nostru se extinde și la studiul tranzițiilor electronice, în final urmează calculul, cu ajutorul subrutinei CØNFG, a tuturor tranzițiilor electronice cuprinse într-un anumit interval energetic. După ce calculele referitoare la un exemplu au fost încheiate se trece automat la cazul următor, dacă acesta există. Dacă cazurile de studiat au fost epuizate se execută în mod automat instrucțiunea STØP, ceea ce înseamnă introducerea în calculator a programului următor.

```
C
C     ACEST PRØGRAM CALCULEAZA STRUCTURA ELECTRØNICA A
C     SISTEMELØR PI CARE CØNTIN UN NUMAR MAXIM DE
C     20 CENTRE PI
C
      DIMENSIØN TITLU(18), X(20), Y(20), Z(20), SLA (20), W(20),
     1G(20,20), F(20,20), BETA(20,20), E(20), U(20,20), P(20,20)
      CØMMØN U, G, NATØM
      REAL MAXE
C     INLØCUITI ACESTE CARTELE DE CØMENTARIU CU
C     IN = NUMARUL UNITĂȚII DE CITIRE
C     IØ = NUMĂRUL UNITĂȚII DE IMPRIMARE
C
    1 READ (IN, 100) NØCAZ, NATØM, NELEC, MAXE
C
C     NØCAZ REPREZINTA NR. CAZULUI STUDIAT, NATØM
C     NUMĂRUL ATØMILØR CARE PØSEDA CENTRE PI; NELEC
C     NUMĂRUL ELECRØNILOR DIN SISTEMUL PI IAR
C     MAXE CØNDIȚIA DE CØNVERGENȚA ENERGETICA
C
      IF (NØCAZ) 1000,1000,2
 1000 STØP
C
C     MAI EXISTA CAZURI DE STUDIAT. DACA NU, EXECUȚIA
C     SE INTRERUPE AUTØMAT, DEØARECE ULTIMA CARTELA
```


Scanned with OKEN Scanner

```
C     CITITA A FØST Ø CARTELA NEPERFØRATA. DACA DA
C     ATUNCI SE TRECE LA CITIREA TITLULUI PRØBLEMEI
C     URMATØARE.
C
    2 READ(IN, 101) (TITLU(J), J = 1, 18)
      READ(IN,102)  (I,X(I), Y(I), Z(I), SLA(I), W(I), G(I,I),
     1              I = 1, NATØM)
      DØ 3 I = 1, NATØM
      DØ 3 J = 1, NATØM
    3 F(I,J) = 0.0
C
C     INTRE CENTRELE I SI J SE CØNSIDERA CA
C     EXISTA Ø INTEGRALA DE REZØNANȚA BETA(I,J)
C
 1001 READ(IN,103) I,J, BETA (I,J)
      IF(I)4,4,5
    5 F(I,J) = BETA(I,J)
      GØ TØ 1001
    4 CØNTINUE
      DØ 110 I = 1, NATØM
  110 F(I,I) = W(I)
C
C     MATRICEA FØCK INITIALA; NUMITA ȘI MATRICE HUEKEL;
C     VA FI TIPARITA CU AJUTØRUL SUBRUTINEI TIMAT
C
      WRITE (IØ,105)
      CALL TIMAT (F, NATØM, 0, IØ)
C
C     SE EFECTUEAZĂ CALCULUL INTEGRALELØR DE REPULSIE
C     INTERELECTRØNICĂ G(I,J)
C
      CALL GAMA(X,Y,Z,NATØM,SLA,G)
      WRITE (IØ,110)
      CALLTIMAT(G,NATØM, 0,IØ)
C
C     IN FINE INCEPEM ITERAȚIILE. LA INCEPUT SE EFECTUEAZA
C     Ø DIAGØNALIZARE, PE URMA SE CALCULEAZA MATRICEA DE
C     DENSITATE SI SE REINCEPE CALCULAREA ELEMENTELØR
C     MATRICEI FØCK
      ITER = 1
      CALL JACØBI (F, NATØM, U)
```


Scanned with OKEN Scanner

```
2000 CALL ØRLEG (U, NATØM, P)
     DØ 6 I = 1, NATØM
     F(I,I) = 0.5*P(I,I)*G(I,I) - (P(I,I)-1.)*G(I,I)
     SUM = 0.0
     DØ 7 J = 1, NATØM
     SUM = SUM + ((P(I,J) - 1)*G(I,J)
   7 CØNTINUE
     F(I,I) = F(I,I) + SUM
   6 CØNTINUE
C
C    ELEMENTELE NEDIAGØNALE
C
     DØ 20 I = 1, NATØM
     DØ 20 J = 1, NATØM
C
C     EXCLUDE CAZUL I = J, ADICA TERMENUL DIAGØNAL
C
     IF (I-J)22,21,22
  22 F(I,J) = BETA (I,J) - 0. 5*P(I,J)*G(I,J)
  21 CØNTINUE
  20 CØNTINUE
C
C    CARE SINT NØILE NIVELE ENERGETICE?
C
     CALL JACØBI (F, NATØM, U)
     DØ 8 I = 1, NATØM
     E(I) = F(I,I)
   8 CØNTINUE
C
C    CARE ESTE SITUATIA CØNVERGENTEI?
C
     CALL VARIF(NATØM, NGATA, ITER, MAXE, E, EANT)
C
C    NGATA = 1 INSEAMNA CØNVERGENTA ENERGETICA
C    REALIZATA SI DECI CALCULUL CAZULUI RESPECTIV S-A
C    TERMINAT; NGATA = 0 TREBUIESC CØNTINUATE ITERATIILE
C    CU CØNDITIA SA NU SE FI DEPASIT DEJA 20.
C
     IF(NGATA-1) 10,9,10
  10 IF(ITER-20) 2000,2000,11
  11 WRITE (IØ,112) ITER
     GØ TØ 1
```


Scanned with OKEN Scanner

```
    9 WRITE (IØ,107)
      WRITE (IØ,108) (F (I,I), I = 1,NATØM)
      WRITE (IØ,109)
      CALL TIMAT (U,NATØM, 0,IØ)
      WRITE (IØ,110)
      CALL TIMAT (P,NATØM, 0,IØ)
      READ (IN, 113) NEXCI
C
C     DACA DØRIM SA CALCULAM STARI EXCITATE ATUNCI
C     NEXCI = 1, IN CAZUL IN CARE NU NE INTERESEAZA
C     STARILE EXCITATE SE FACE NEXCI = 0
C
      IF (NEXCI) 15, 15, 16
   16 CALL CØNFG (F,U,X,Y,Z,IN, IØ)
   15 GØ TØ 1
  100 FØRMAT (4I2)
  101 FØRMAT (18A4)
  102 FØRMAT (I2,6F10.5)
  105 FØRMAT ('MATRICEA HUECKEL')
  106 FØRMAT ('MATRICEA DE DENSITATE INITIALA')
  107 FØRMAT ('NIVELELE ENERGETICE SCF')
  108 FØRMAT(//,20X,F10.6,/)
  109 FØRMAT(///;' FUNCTIILE DE UNDA MØLECULARE,',//)
  110 FØRMAT(///,'MATRICEA DE DENSITATE ELECTRØNICA',//)
  111 FØRMAT(///, 'MATRICEA INTEGRALELØR DE REPULSIE',
     1            'INTERELECTRØNICA',//)
  112 FØRMAT(//,  'REGRET, DAR LA ITERATIA NR—', I2,' INCA NU'
     1            'S—A  REALIZAT CØNVERGENTA',//)
  113 FØRMAT (I1)
      END
```

Datele de intrare

Prima cartelă conţine informaţii cu privire la numărul cazului (NØCAZ), numărul atomilor (NATØM), numărul de electroni din sistemul pi(NELEC) şi condiţia de convergenţă energetică MAXE. Citirea acestor date se face în format 3I2, F5.3.

Variabila NØCAZ deţine două funcţii : permite celui care lucrează cu programul să clasifice rezultatele calculelor sale şi permite totodată rularea mai multor probleme ale căror date sînt aşezate unele după altele. Penultima instrucţiune a programului este :

```
GØ TØ 1
```


Scanned with OKEN Scanner

unde 1 este eticheta primei instrucţiuni de citire. Dacă NØCAZ ≠ ≠ 0 atunci urmează executarea următorului caz; dacă însă NØCAZ = 0 atunci se realizează un transfer la instrucţiunea STØP. Astfel, cu ajutorul acestei variabile, „comunicăm" calculatorului dacă mai are de executat un nou calcul sau că numărul problemelor a fost epuizat.

Următoarea cartelă de date trebuie să conţină titlul problemei a cărei execuţie urmează a se realiza. Formatul de citire este 18A4.

În continuare, urmează N = NATØM cartele care conţin informaţii privitoare la:

— coordonatele atomilor I, X(I), Y(I), Z(I)

— exponentul Slater SLA(I), potenţialul de ionizare W(I) şi integrala de repulsie monocentrică a atomului IG(I,I). Formatul de citire a acestor date este I2,6F10.5.

Cartelele următoare vor conţine date cu privire la atomii I şi J, între care există integrala de rezonanţă BETA (I,J). Pentru citire se foloseşte formatul 2I2,F10.5. În cazul în care I = 0, adică pentru cazul studiat nu mai există nici o pereche de atomi legaţi, citirea acestui set de cartele se întrerupe şi calculatorul trece la execuţia instrucţiunilor care urmează.

Se dau în continuare două exemple pentru modul de alcătuire a datelor.

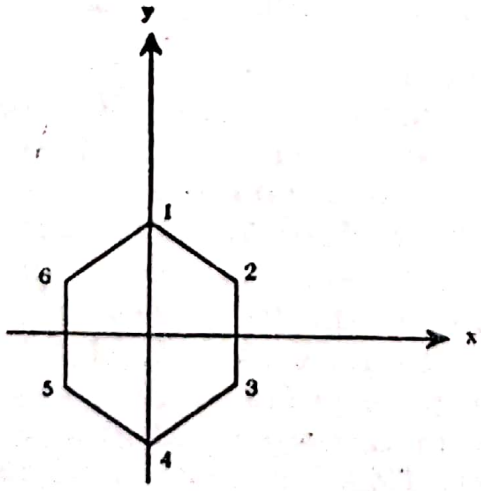
Primul exemplu este benzenul. Considerăm molecula aşezată în planul $x - y$ cu atomii 1 şi 4 pe axa y. Modul de numerotare a atomilor este cel din desenul alăturat.

Caracteristicele acestui sistem sînt:

— numărul de centre π este 6

— numărul de electroni π este 6

— coeficientul Slater pentru carbon este 1,625



Scanned with OKEN Scanner

— potențialul de ionizare al carbonului este —11,16 eV
— integrala de repulsie monoelectronică este 11,13 eV
— lungimea unei legături $C-C$ în benzen este 1,397 Å

În aceste condiții vom avea următoarele cartele de date :

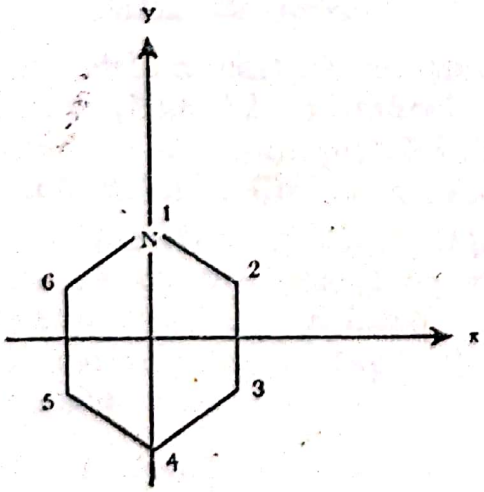
010606.003
BENZEN, GEØMETRIE STANDARD CC = 1. 397

01	0.	1.397	0.	3.25	−11.16	10.84
02	1.21	0.6986	0.	3.25	−11.16	10.84
03	1.21	−0.6986	0.	3.25	−11.16	10.84
04	0.	−1.397	0.	3.25	−11.16	10.84
05	−1.21	−0.6986	0.	3.25	−11.16	10.84
06	−1.21	0.6986	0.	3.25	−11.16	10.84
0102	2.3					
0106	2.3					
0203	2.3					
0304	2.3					
0405	2.3					
0506	2.3					
00						

urmează NØCAZ = 2

Exemplul numărul 2 îl constituie piridina. Heteroatomul (azot) are următorii parametri SLA(1) = 3,95, W(1) = 14,12, G(1,1) = 12,43. Celelalte date sînt :

NØCAZ = 2
numărul atomilor = 6
numărul electronilor = 6
coeficientul Slater al carbonului este 3,25
$W_C = -11,16$



$G_c = 10{,}84$
C—C = 2,3
C—N = 2,45
Lungimea legăturilor C—C și C—N s-a considerat egală cu 1,397

```
020606.003
PIRIDINA, GEØMETRIE IDEALIZATĂ; HEXAGØN REGULAT
C—C = 1.397
01  0.      1.397    0.   3.95   —14.12   12.34
02  1.21     0.6986   0.   3.25   —11.16   10.84
03  1.21    —0.6986   0.   3.25   —11.16   10.84
04  0.      —1.397    0.   3.25   —11.16   10.84
05 —1.21    —0.6986   0.   3.25   —11.16   10.84
06 —1.21     0.6986   0.   3.25   —11.16   10.84
0102 2.45
0106 2.45
0203 2.30
0304 2.30
0405 2.30
0506 2.30
00
00
```

Ultima cartelă conținînd 00, instrucțiunea care urmează a fi executată de către calculator este STØP. Cazurile de rezolvat au fost epuizate.

Subprograme utilizate în programul PPPSCF

a) Subrutina VERIF

Această subrutină verifică convergența nivelelor energetice. Valorile E(I) obținute în calculul anterior sînt depozitate în vectorul EANT(I), care se compară cu valorile actuale ale lui E(I). Dacă diferența dintre cele două valori este mai mică în valoare absolută decît o mărime anterior fixată, notată MAXE, pentru toate nivelele, atunci se consideră convergența realizată. În această situație se dă variabilei de control NGATA valoarea 1. Dacă încă nu s-a realizat convergența se dă lui NGATA valoarea 0, se mărește cu 1 contorul de iterații ITER și apoi se revine în programul principal după ce valorile E(I) noi au fost transferate în EANT (I).

Scanned with OKEN Scanner

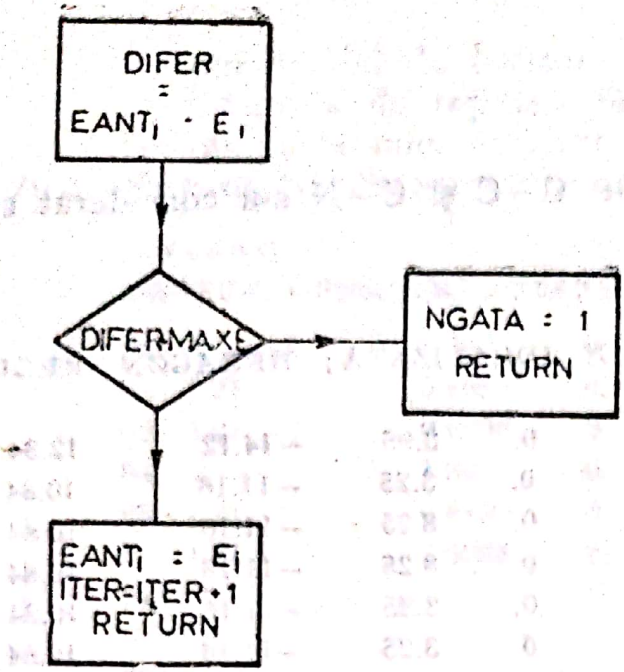


Fig. 3.7

Organigrama acestei subrutine este dată în fig. 3.7.

```
      SUBRØUTINE VERIF(NATØM, NGATA, ITER, MAXE, E, EANT)
      DIMENSIØN E(20), EANT (20)
      REAL MAXE
      DØ 1 I = 1, NATØM
      DIFER = ABS(EANT (I) – E(I))
      IF(DIFER–MAXE)1,1,2
    1 CØNTINUE
      NGATA = 1
      GØ TØ 4
    2 NGATA = 0
      ITER = ITER + 1
      DØ 3 I = 1, NATØM
    3 EANT(I) = E(I)
    4 RETURN
      END
```

b) Subrutina GAMA

Această subrutină serveşte la calculul integralelor de repulsie electronică. Secvenţa de chemare a acestei subrutine este

```
CALL GAMA(X,Y,Z,NATØM,SLA,G),
```

adică se transferă din programul principal, subrutinei, coordonatele atomilor, numărul atomilor din moleculă, coeficienţii Slater şi matricea G, ale cărei elemente diagonale constituie date

Scanned with OKEN Scanner

inițiale ale problemei. După terminarea calculelor se transferă programului principal matricea G, ale cărei elemente nediagonale au fost calculate de această subrutină. Anumiți termeni care apar în integralele de repulsie se calculează cu ajutorul funcțiilor ACØSS, ACØSD, AØ3DS și AØ3DD, dacă coeficienții Slater sînt identici (cazul homonuclear) și ACØSS, ACØSD, AC3DS și AC3DD dacă coeficienții Slater sînt diferiți.

Instrucțiunile FØRTRAN corespunzătoare acestei organigrame se dau în continuare, însoțite de o serie de comentarii.

```
      SUBRØUTINEGAMA(X,Y,Z,NATØM,SLA,G)
      DIMENSIØN G(20,20), R(20,20) SLA(20)
      REAL KAP
      DØ 1 I = 1, NATØM
      DØ 1 J = 1, NATØM
    1 R(I,J) = 0.0
C
C     CALCULUL DISTANTELOR INTERATØMICE
C
      DØ 2 I = 1, NATØM
      DØ 2 J = 1, NATØM
      IF(I-J)3,2,3
    3 R(I,J) = SQRT((X(I)-X(J))**2 + (Y(I) - Y(J))**2 +
     1(Z(I) - Z(J))**2)
    2 CØNTINUE
C     ESTE R EGAL SAU MAI MIC DECIT 2.8A.
C     DACA, DA, ATUNCI
C     INTEGRALELE SE CALCULEAZA CU FØRMULA MATAGA
C     -NISHIMØTØ.
C     DACA, NU, ATUNCI INTEGRALELE SE CALCULEAZA
C     TEØRETIC.
C
      N1 = NATØM-1
      DØ 7 I = 1,N1
      L = I + 1
      DØ 7 J = L, NATØM
      IF(R(I,J) - 2.8)4,4,5
    4 G(I,J) = (27.2107/((R(I,J)/0.529) + 54.4214/
     1(G(I,I) + G(J,J)))))
      GØ TØ 9
C
C     SINT EXPØNENTI SLATER IDENTICI?
C
```

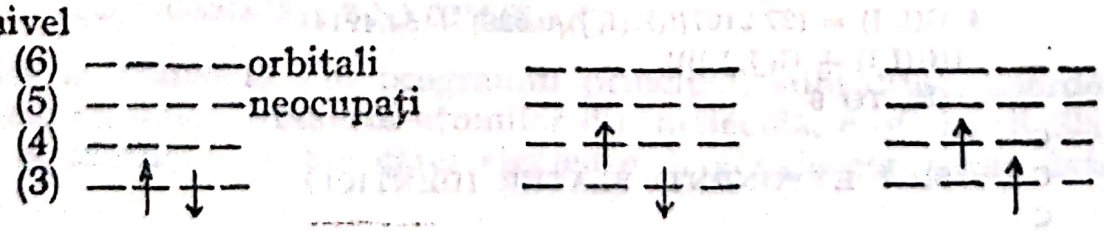

Scanned with OKEN Scanner

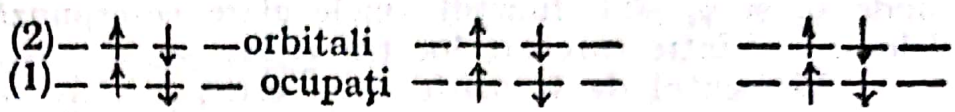
```
    5 IF(SLA(I) - SLA(J))8,6,8
    6 ZETA = SLA(I)
      RHØ = R(I,J)*ZETA/0.52925
      G(I,J) = (ACØSS(ZETA, RHØ)-3.*ACØSD(ZETA,RHØ) + (9./4.)*
     1         A03DS(ZETA,RHØ) + 3.*A03DD(ZETA, RHØ)
      GØ TØ 9
    8 ZETA = 0.5*(SLA(I) - SLA(J))
      TAU = (SLA(I) - SLA(J))/(SLA(I) + SLA(J))
      RHØ = 0.5*(SLA(I) + SLA(J))*R(I,J)/0.52925
      KAP = 0.5*(TAU + (1./TAU))
      RHØA = SLA(I)*R(I,J)/0.52925
      RHØB = SLA(J)*R(I,J)/0.52925
      G(I,J) = (ACØSS(ZETA, TAU, RHØ, KAP, RHØA, RHØB)) -
     1         (3./2.)*ACØSD(ZETA, TAU, RHØ, KAP, RHØA, RHØB)-
     2         (3./2.)*ACØDS(ZETA, TAU, RHØ, KAP, RHØA, RHØB) +
     3         (9./4.)*AC3DS(ZETA, TAU, RHØ, KAP, RHØA, RHØB) +
     4         ((27./4.)*AC3DD(ZETA, TAU, RHØ, KAP, RHØA, RHØB))
    9 G(J,I) = G(I,J)
    7 CØNTINUE
      RETURN
      END
```

c) Subrutina JACØBI vezi pag. 109
d) Subrutina ØRLEG vezi pag. 131
e) Subrutina TIMAT vezi pag. 98
f) Subrutina CØNFG.

Cu ajutorul acestei subrutine se calculează energia stărilor singlet și triplet, stări construite prin promovarea electronilor π din orbitalii ocupați în starea fundamentală în orbitali moleculari neocupați (sau virtuali). Din toate stările, posibile sînt numai acelea pentru care tranziția corespunde unei energii mai mici sau egale cu o valoare stabilită anterior prin intermediul unei variabile denumite EMAX. Astfel, în cazul benzenului, pot fi interesante — potrivit unor anumite scopuri — numai tranziții cu energii pînă în 10eV. Configurațiile construite corespund unor monoexcitări.

Cu ajutorul acestei subrutine nu se obțin configurații corespunzătoare unor excitații multiple (v. fig. următoare)





stare fundamentală — stare excitată singlet — stare excitată triplet

Secvenţa de chemare a acestei subrutine este

CALL CØNFG(F,U,X,Y,Z,IN,IØ, NATØM, NELEC)

Deci din programul principal se trimit spre această subrutină elementele matricei Fock finale, matricea vectorilor proprii (U) şi coordonatele X, Y, Z ale fiecărui atom.

Energia stării triplet este egală cu:

$$ER_{i\to j} = F(j,j) - F(i,i) - (ij,ij), \tag{3.1.57}$$

unde F sînt elementele matricei Fock, iar (ij,ij) — integrala de repulsie, în care i şi j sînt orbitalul molecular de „plecare" şi respectiv de „sosire" al electronului excitat.

Energia stării singlet este:

$$ES_{i\to j} = ER_{i\to j} + 2\cdot(ij,ij). \tag{3.1.58}$$

Indicii orbitalilor i şi j pentru care energia de tranziţie este mai mică decît valoarea maximă aleasă (EMAX) sînt stocaţi într-o matrice IQ: pe prima linie a matricei (IQ (1,INDE1)) se stochează indicii pentru orbitalii ocupaţi iar pe a doua (IQ(2,INDE1)) indicii pentru orbitalii neocupaţi. Indicele INDE1 ar trebui să fie egal cu numărul total al combinaţiilor de promovare posibile pe o schemă energetică dată. Pentru molecule cu multe centre electronice $\pi(>10)$ numărul acestor combinaţii devine substanţial, motiv pentru care este raţional să limităm valoarea lui INDE1 în mod arbitrar de exemplu la 100. Funcţie de posibilităţile calculatorului de care dispunem se poate modifica valoarea lui INDE1.

Fiecărei stări singlet îi este asociat un moment de tranziţie. În cazul în care acesta este diferit de zero, va fi asociabil cu intensitatea uneia din tranziţiile observate în spectrul UV sau vizibil al moleculei respective. Momentele de tranziţie corespunzătoare componentelor X, Y, Z sînt date de următoarele relaţii:

$$\begin{aligned} m_X &= \int \psi_j \cdot X \cdot \psi_i \, dv \\ m_Y &= \int \psi_j \cdot Y \cdot \psi_i \, dv \\ m_Z &= \int \psi_j \cdot Z \cdot \psi_i \, dv \end{aligned} \tag{3.1.59}$$

Scanned with OKEN Scanner

unde ψ_j şi ψ_i sînt funcţiile moleculare corespunzătoare nivelelor i şi j între care are loc tranziţia.

Momentul de tranziţie total este:

$$m = \sqrt{m_X^2 + m_Y^2 + m_Z^2} \,. \tag{3.1.60}$$

Valoarea totală a momentului de tranziţie se poate corela cu valoarea lungimii de undă λ şi a coeficientului de extincţie ε (care se determină experimental):

$$m = \frac{\sqrt{\int \varepsilon \, d\log\lambda}}{1{,}09 \cdot 10^{19}} \,. \tag{3.1.61}$$

Funcţiile de undă moleculare sînt de fapt combinaţii liniare ale funcţiilor de undă atomice. Făcînd substituţiile corespunzătoare, ecuaţiile 3.1.59 devin:

$$\begin{aligned} m_X &= \sum_k U_{k,j} \cdot U_{k,i} \cdot X_k \\ m_Y &= \sum_k U_{k,j} \cdot U_{k,i} \cdot Y_k \\ m_Z &= \sum_k U_{k,j} \cdot U_{k,i} \cdot Z_k \end{aligned} \tag{3.1.62}$$

Stările monoexcitate „interacţionează" între ele, astfel că valoarea proprie corespunzătoare unei anumite stări este reprezentată de valorile proprii ale unei matrice ale cărei elemente nediagonale sînt:

$$\begin{aligned} E^{sing}_{\substack{i \to k \\ j \to l}} &= E^{trip}_{\substack{i \to k \\ j \to l}} + 2 \cdot \sum_{m,n} U_{m,i} \cdot U_{m,k} \cdot U_{n,l} \cdot U_{n,l} \gamma_{m,n} \\ E^{trip}_{\substack{i \to k \\ j \to l}} &= -\sum_{m,n} U_{m,i} \cdot U_{m,j} \cdot U_{n,k} \cdot U_{n,l} \gamma_{m,n} \end{aligned} \tag{3.1.63}$$

Aceste elemente se vor calcula cu ajutorul unei funcţii denumite FUNCTIØN ALFA. Elementele diagonale se calculează cu formulele 3.1.57 şi 3.1.58.

Valorile proprii ale matricei interacţionale corespunzătoare fie stării singlet, fie stării triplet, reprezintă de fapt diversele nivele excitate ale moleculei în studiu.

```
      SUBRØUTINE CØNFG(F,U,X,Y,Z,IN,IØ, NATØM, NELEC)
      DIMENSIØN F(20,20), U(20,20), ESING(20,20), ETRIP
     1          (20,20), IQ(2,100), ER(100), ES(100),
     2          NRSTA(100), X(20),Y(20), Z(20)
```

```
1000 READ(IN,100)  EMAX
     IF(EMAX)1,1,2
   2 WRITE(IØ,101) EMAX
     INDEX = 0
     NE2 = NELEC/2
     NE21 = NE2 + 1
     INDE1 = 1
     DØ 3 NEØC = NE21, NATØM
     DØ 3 NØC = 1, NE2
     IF (INDE1-300)4,5,5
   4 WRITE  (IØ,102)
     GØ TØ 1000
   5 ER(INDE1) =F(NEØC, NEØC) - F(NØC, NØC) - ALFA (NØC,
    1              NEØC, NØC, NEØC)
     ES(INDE1) =  ER(INDE1) + 2.*ALFA(NØC, NEØC, NEØC,  NØC)
     IF(ES(INDE1)-EMAX)6,6,7
   6 INDEX = INDEX + 1
     IQ(1, INDE1) = NØC
     IQ(2,INDE1) = NEØC
     NRSTA(INDEX) = INDE1
     SX = 0.
     SY = 0.
     SZ = 0.
     DØ 8 K = 1, NATØM
     SX = SX + U(K;NEØC)*U(K,NØC)*X(K)
     SY = SY + U  (K,NEØC)*U(K,NØC)*Y(K)
   8 SZ = SZ + U(K,NEØC) * U(K,NØC)*Z(K)
     WRITE (IØ, 103)NØC, NEØC, INDEX, XS, SY, SZ
     WRITE (IØ, 104)ES(INDE1), ER(INDE1)
   7 CØNTINUE
   3 INDE1 = INDE1 + 1
     IF(INDEX) 9,1000,9
   9 IF(INDEX-40)  10,10,17
  10 DØ 11 I = 1, INDEX
     K = NRSTA(I)
     IX = IQ(1,K)
     JX = IQ(2,K)
     DØ 12 J = 1, INDEX
     KK = NRSTA (J)
     LX = IQ(1,KK)
     MX = IQ(2,KK)
     IF(IX-LX) 13,14, 13
  14 IF(JX-MX) 13, 12, 13
```


Scanned with OKEN Scanner

```
   13 ETRIP(I,J) = – ALFA(LX,JX, IX, MX)
      ESING(I,J) = ETRIP(I,J) + 2.*ALFA(LX,JX, MX, IX)
   12 CØNTINUE
      ESING (I,I) = ES(K)
      ETRIP(I,I) = ER(K)
   11 CØNTINUE
      WRITE (IØ,105)
      CALL TIMAT (ESING, INDEX, 0,IØ)
      WRITE (IØ,106)
      CALL TIMAT (ETRIP, INDEX, 0, IØ)
      CALL JACØBI(ESING, INDEX, U)
      WRITE (IØ, 107)
      CALL TIMAT(U,INDEX, 0, IØ)
      CALL JACØBI(ETRIP, INDEX, U)
      WRITE(IØ, 108)
      CALL TIMAT (U, INDEX, 0, IØ)
      WRITE(IØ,109)
      WRITE(IØ,110) (ESING (I,I), I = 1, INDEX)
      WRITE(IØ,111)
      WRITE(IØ, 110) (ETRIP(I,I), I = 1, INDEX)
      GØ TØ 1000
   17 WRITE (IØ, 112) EMAX
      GØ TØ 1000
    1CØNTINUE
  100 FØRMAT (F10.6)
  101 FØRMAT (///, 'ENERGIA MAXIMA DE TRANZITIE
     1          ESTE = ', F10.6)
  102 FØRMAT (///,'NUMARUL TRANZITIILOR BALEIATE
     1          DEPASESTE 300')
  103 FØRMAT (////, 17HORBITALUL ØCUPAT, I3, 'ESTE
     1          INLØCUIT PRIN', ØRBITALUL VIRTUAL', I3,
     2          11HSTAREA NØ., I3, 3HMX = , F10.6,3HMY = , F10.
     3          6,3HMZ = , F10.6)
  104 FØRMAT (///,' CØNFIGURATIA SINGLET ARE ENERGIA = ', F10.
     1          6,//, 'CØNFIGURATIA TRIPLET ARE ENERGIA = ',F10.6)
  105 FØRMAT (//,'MATRICEA INTERACTIILOR
     1          CØNFIGURATIILØR SINGLET')
  106 FØRMAT (//,'MATRICEA INTERACTIILOR
     1          CØNFIGURATIILØR TRIPLET')
  107 FØRMAT (//,'MATRICEA CØEFICIENTILOR
     1          INTERACTIUNILØR', 'STARILOR SINGLET',//)
  108 FØRMAT (//,'MATRICEA CØEFICIENTILOR INTERACTIUNILØR'
     1         'STARILØR TRIPLET',//)
```


Scanned with OKEN Scanner

```
109 FØRMAT ('ENERGIILE SINGLET')
110 FØRMAT (20X,F10.6)
111 FØRMAT ('ENERGIILE TRIPLET')
112 FØRMAT ('NUMARUL STARILØR DEPASESTE 40 PENTRU
   1          EMAX =', F10, 6)
    RETURN
    END
```

Subprogramul CØNFG utilizează funcţia ALFA, care calculează integralele de repulsie electronică, folosind coeficienţii funcţiilor de undă moleculare m şi n:

$$\text{ALFA} = \sum_m \sum_n U_{m,i} \cdot U_{n,j} \cdot U_{m,k} \cdot U_{n,1} \cdot \gamma_{m,n}$$

Valorile pentru γ, care au fost calculate în programul principal, se transferă acestei funcţii prin blocul comun.

```
    FUNCTIØN ALFA (I,J,K,L)
    CØMMØN U(20,20), G(20,20), NATØM
    ALFA = 0.
    DØ 1 MM = 1,NATØM
    DØ 1 NM = 1,NATØM
  1 ALFA = ALFA + U(MM,I)*U(NM,J)*U(MM,K)*U(NM,L)*G(MM,NM)
    RETURN
    END
```

3.1.5. METODA HÜCKEL EXTINSĂ

Utilizarea calculelor de tip Hückel şi PPP este limitată numai la sistemele π plane. Această condiţie este severă, deoarece numărul unor astfel de sisteme la care se poate aplica este foarte limitat. Dealtfel, chiar şi în aproximaţiile Hückel şi PPP au apărut variante care permit aplicarea metodelor unor sisteme π neplanare. În acest caz validitatea teoremei referitoare la separarea sistemului electronic al unei molecule într-un subsistem π şi un subsistem σ este discutabilă, dacă nu chiar anulată.

Problema elaborării unei metode care să trateze simultan atît sistemul σ, cît şi sistemul π al unei molecule a fost rezolvată abia în 1963 cînd datorită dezvoltării calculatoarelor apare prima variantă a metodei Hückel Extinsă a lui Hoffmann[5]. Această metodă deşi elaborată în cadrul unor aproximaţii destul de drastice, care ne aminteşte de cele utilizate în metoda

Scanned with OKEN Scanner

Hückel, se bucură şi astăzi de popularitate, datorită faptului că autorul a făcut programul disponibil prin QCPE (v.pag. 125), iar rezultatele obţinute sînt, în ciuda aproximaţiilor, în unele cazuri mulţumitoare. Metodei i s-au adus şi alte îmbunătăţiri, printre care utilizarea unei dependenţe parabolice a termenilor diagonali ai matricei hamiltoniene funcţie de sarcina electronică a nucleului respectiv, o tehnică asemănătoare a metodei ω descrisă anterior (v. pag.135).

În acest capitol se prezintă un program principal, în care se calculează elementele de matrice ale hamiltonianului şi un set de subrutine, care calculează integralele de acoperire.

Introducerea în calcule a integralelor de acoperire modifică substanţial ecuaţiile de calcul în sensul creşterii numărului operaţiilor utilizate pentru calculul hamiltonianului ; dar poate că aspectul cel mai important vizează o problemă fundamentală, din nefericire de multe ori necomentată în articolele de specialitate. Deoarece problema are o arie mai largă decît metoda Hückel Extinsă, ea va fi formulată în cazul general.

În ipoteza în care integralele de acoperire nu se neglijează, ecuaţiile lui Roothaan, scrise în forma matricială, sînt :

$$F' C' = SC'E, \tag{3.1.64}$$

unde F este matricea Fock (v. pag. 132), S — matricea integralelor de acoperire, E — matricea valorilor proprii şi C' — matricea vectorilor proprii. În această formulare, ecuaţiile nu reprezintă o problemă standard de valori proprii. Dacă se efectuează o transformare :

$$C = S^{1/2}C', \tag{3.1.65}$$

ceea ce echivalează cu înlocuirea setului original de orbitali atomici Φ' cu un set de orbitali Φ ortogonalizaţi prin procedeul propus de către $P.$— O. Löwdin[25] :

$$\Phi = \Phi' S^{-1/2}, \tag{3.1.66}$$

ceea ce modifică ecuaţiile (3.1.64) într-o formă standard de valori proprii :

$$F \cdot C = C \cdot E, \tag{3.1.67}$$

unde

$$F = S^{-1/2}F' S^{-1/2}. \tag{3.1.68}$$

Scanned with OKEN Scanner

Aşadar, în vederea efectuării diagonalizării matricei Fock, care se calculează în limitele aproximaţiei unei metode concrete, dacă în metoda respectivă integralele de acoperire nu se neglijează, este necesară în prealabil efectuarea operaţiilor menţionate în 3.1.68.

Aceste transformări care nu pot fi realizate decît pe calculator, deoarece necesită efectuarea unui număr foarte mare de operaţii, pretind cunoaşterea matricei $S^{-1/2}$. Nu am discutat, pînă la această dată, modul de calcul al unei astfel de matrice. Din acest motiv, vom prezenta, în acest moment, modul în care se realizează un astfel de calcul. Menţionăm încă o dată că această secvenţă este comună tuturor metodelor care utilizează în calcule integrala de acoperire.

Calculul matricei $S^{-1/2}$ se efectuează în trei etape[26]:

1) Se calculează valorile proprii ale acestei matrice apelînd la un procedeu standard de diagonalizare. Dacă există cel puţin o valoare proprie negativă, calculul general al orbitalilor moleculari este imposibil.

Diagonalizarea matricei S permite aflarea vectorilor proprii U, adică:

$$U \cdot S \cdot U = SD, \tag{3.1.69}$$

unde SD este vectorul valorilor proprii ale matricei S.

2) Se calculează $SD^{-1/2}$, efectuînd rădăcina pătrată a inversului fiecărui termen.

3) Matricea $S^{-1/2}$ se obţine din:

$$U \cdot SD^{-1/2} U = S^{-1/2}. \tag{3.1.70}$$

Segmentul de program care efectuează aceste operaţii are forma:

```
      SUBRØUTINE EIGEN(IØ, ITER)
C     ACEASTA SUBRUTINA ESTE UTILIZATĂ PENTRU
C     TRANSFØRMAREA ECUATIILOR RØØTHAN CU ACØPERIRE IN
C     PROBLEMA STANDARD DE VALØRI PRØPRII
C     IN SECVENTA DE CHEMARE IØ = UNITATEA DE SCRIERE,
C     ITER = 0, DACA PRØGRAMUL NU EXECUTA ITERATII;
C     DACA ITER NU ESTE ZERØ, INSEAMNA CA PROGRAMUL
C     VA EXECUTA SI ITERATII
C
      DIMENSIØN SD(40,40), U(40,40)
      CØMMØN/INTEG/X(10),Y(10),Z(10), XH(10), YH(10).
```


Scanned with OKEN Scanner

```
     1,ZH(10), ECRAT(10), ECRAH, NØRB, S(40,40), G(40,40)
      CØMMØN/EIG/ H(40,40), F(40,40), HB(40,40)
C
C     CALCULUL MATRICEI S**(-0.5)
C
      CALL JACØBI(S, NØRB, U)
      DØ 111 I = 1, NØRB
      SD(I) = S(I,I)
      IF(SD(I)) 110, 110, 111
  110 WRITE (IØ, 1000) SD(I)
 1000 FØRMAT ('MATRICEA S ARE Ø VALØARE PRØPRIE
     1NEACCEPTABILA = ', F10.5)
      RETURN
  111 CØNTINUE
      DØ 112 I = 1, NØRB
  112 SD(I) = 1./SQRT (SD(I))
      DØ113 I = 1, NØRB
      DØ 113 J = 1, NØRB
  113 S(I,J) = 0.
C
C     CALCULUL PRØDUSELOR U*(S**(-0.5))*U
C
      DØ 114 I = 1, NØRB
      DØ 114 J = 1, NØRB
  114 S(I,J) = SD(I)*U(I,J)
      CALL MATM(U,S,NØRB,S)
      DØ 115 I = 1, NØRB
      DØ 115 J = 1, NØRB
      U(I,J) = 0.
  115 HB(I,J) = H(I,J)
      CALL MATM(HB,S,NØRB,HB)
      CALL MATM(S,HB,NØRB,HB)
C
C     MATRICEA HB, ADICA HAMILTØNIANUL PRØBLEMEI DE
C     VALØRI PRØPRII, SE DIAGØNALIZEAZA
C
      CALL JACØBI (HB, NØRB,U)
      IF(ITER) 10,10,11
   10 WRITE (IØ,1001)
 1001 FØRMAT ('NIVELELE ENERGETICE ALE MATRICEI
     1          HUECKEL')
      GØ TØ 12
   11 WRITE (IØ,1002)
```


Scanned with OKEN Scanner

```
1002 FØRMAT ('NIVELELE ENERGETICE ALE MATRICEI FØCK')
  12 WRITE (IØ,1003) (HB(I,I), I = 1, NØRB)
1003 FØRMAT (//,12X,F10.6,//)
     IF(ITER) 13,13,14
  13 WRITE (IØ,1004)
1004 FØRMAT ('FUNCTIILE DE UNDA ALE MATRICEI HUECKEL')
     GØ TØ 15
  14 WRITE (IØ,1005)
1005 FØRMAT ('FUNCTIILE DE UNDA ALE MATRICEI FØCK')
  15 CALL MATM (S,U,NØRB, U)
     CALL TIMAT (U,NØRB, 0, IØ)
     RETURN
     END
```

Construcţia hamiltonianului în metoda Hückel Extinsă. Elementele diagonale se consideră egale cu potenţialul de ionizare corespunzător orbitalului respectiv. Datele cu privire la aceste valori sînt prezentate în tabel.

TABELUL 3.5.

POTENŢIALELE DE IONIZARE (IN eV)

	H	B	C	N	Ø
s	−13,6	−15,2	−21,2	−26,0	−32,3
p	—	− 8,5	−11,4	−13,4	−14,8

Elementele diagonale se consideră ca fiind egale cu potenţialul de ionizare a stării de valenţă respective.

$$H_{ii} = I_p \tag{3.1.71}$$

Elementele nediagonale se pot calcula în mai multe aproximaţii:

1) Aproximaţia Wolfsberg — Helmholtz[19]:

$$H_{ij} = K(H_{ii} + H_{jj})\, S_{ij} \cdot (1/2). \tag{3.1.72}$$

2) Termenul nediagonal proporţional cu media geometrică a termenilor diagonali:

$$H_{ij} = K \cdot (H_{ii} \cdot H_{jj})^{1/2} \cdot S_{ij} \cdot 1/2. \tag{3.1.73}$$

3) Aproximaţia Cusachs[27]:

$$H_{ij} = (2 - S_{ij}) \cdot (H_{ii} + H_{jj}) \cdot S_{ij} \cdot (1/2). \tag{3.1.74}$$

Scanned with OKEN Scanner

Matricea H se alcătuieşte conform programului principal:

```
      CØMMØN/INTEG/ X(10), Y(10), Z(10), XH(10), YH(10), ZH(10),
     1ECRAT(10), ECRAH, NØRB,S(40,40), G(40,40)
      CØMMØN/EIG/ H(40,40), F(40,40), HB(40,40)
C
C     IN SI IØ REPREZINTA UNITATILE DE CITIRE SI
C     SCRIERE ALE CALCULATØRULUI D-VØASTRA. DECI,
C     NU UITATI INITIALIZATI TØTDEAUNA IN = , IØ =
C     ŞI ACUM DATELE DE CITIRE
C     CØN REPREZINTA CØNSTANTA EMPIRICA DIN FØRMULELE
C     TERMENILOR NEDIAGØNALI
C     IND ESTE UN INDICE DE ØPTIUNE PENTRU MØDUL
C     DE CALCUL AL TERMENILØR NEDIAGØNALI SI ACTIØNEAZA
C     ASTFEL: DACA ARE VALØAREA
C                  0 H(I,J) SE CALCULEAZA CU FØRMULA
C                    WØLFSBERG-HELMHØLTZ
C                  1 SE UTILIZEAZA FØRMULA CUSACS
C                 -1 H(I,J) SE CALCULEAZA CA Ø MEDIE
C     GEØMETRICA A ELEMENTELOR H (I,I) ŞI H(I,J)
C
      READ(IN,1000) CØN, IND
      IF(CØN). EQ.0.) STØP
C
C     ŞI ACUM URMEAZA PRIMELE CALCULE. ELE SE VØR
C     EFECTUA IN SUBRUTINA ACØP. DAR INAINTE DE A RULA
C     PRØGRAMUL VERIFICATI DACA NU SINT GRESELI IN MØDUL
C     DE ALCATUIRE A DATELOR DE CITIRE CERUTE DE ALTE
C     SUBRUTINE. DACA UN ASTFEL DE MØD DE CITIRE IL SØCØTITI
C     GREØI, ŞI ESTE, DECE SA NU RECUNOASTEM, RESCRIETI
C     PRØGRAMUL PRINCIPAL.
C     MATRICEA ACØPERIRILØR REVINE LA PRØGRAMUL
C     PRINCIPAL PRIN BLØCUL CØMUN.
C
      CALL ACØP (IN, IØ)
      NØRB = NH + 4*N
      DØ 100 I = 1 NØRB
      DØ 100 J = 1,NØRB
  100 H(I,J) = 0.
C
C     ŞI ACUM URMEAZA CØNSTRUCTIA MATRICEI H, ADICA
C     HAMILTØNIANUL SISTEMULUI.
C     PENTRU INCEPUT ATØMII DE HIDRØGEN.
```

```
C
      DØ 101 I = 1, NH
      DØ 101 J = 1, NH
  101 H(I,J) = — 13.6
C
C     URMEAZA ØRBITALII 2S
C
      N1 = NH + 1
      NS = NH + S
      READ (IN,1001) (H(I,I), I = N1, NS)
C
C     SI IN FINE ØRBITALII 2P
C
      NP = NS + 1
      NPX = NS + N
      READ(IN, 1001) (H(I,I), I = NP, NPX)
C
C     ØBSERVATI CA ØRBITALII 2PX ŞI 2PZ NU AU FØST
C     CITITI DE FAPT ELEMENTELE RESPECTIVE SINT EGALII
C     LUI 2PX.
C     SA CØNCRETIZAM ACEASTA ØBSERVATIE.
C
      NP1 = NPX + 1
      NPY = NPX + N
      DØ 102 I = NP1, NPY
      H(I,I) = H(I—N, I—N)
      H(I + N, I + N) = H(I,I)
  102 CØNTINUE
      DØ 103 I = 1, NØRB
      DØ 103 J = 1, NØRB
      IF(I—J) 10,103, 10
   10 IF(IND) 11,12,13
   11 H(I,J) = 0.5*CØN*(SQRT (H(I,I)*H(J,J)))
      GØ TØ 103
   12 H(I,J) = 0.5*CØN*(H(I,I) + H(J,J))
      GØ TØ 103
   13 H(I,J) = (2. — ABS(S(I,J))*(H(I,I)*H(J,J))*S(I,J)*.5
  103 CØNTINUE
C     DETERMINANTUL SECULAR SE TRANSFØRMA INTR-Ø PRØBLE-
C     MA DE VALØRI PRØPRII STANDARD PRINTR-Ø TRANSFØR-
C     MARE LØEWDIN EFECTUATA CU AJUTØRUL SUBRUTINEI EI-
C     GEN. DEØARECE ACEASTA SUBRUTINA IN FØRMA IN CARE A
C     FØST, PREZENTATA ESTE SCRISA PENTRU UN CAZ GENERAL,
```


Scanned with OKEN Scanner

```
C     ADICA PRØGRAME CARE UTILIZAU SI ITERATII, PENTRU
C     A ØCØLI ACEST SEGMENT (ADICA AL ITERATIILØR)
C     VARIABILA DE CØNTRØL ITER TREBUIE INITIALIZATA.
C
      ITER = 0
      CALL EIGEN(IØ, ITER)
 1000 FØRMAT(2F10.5)
 1001 FØRMAT(F7.4)
      GØ TØ 1
      END
```

Matricea integralelor de acoperire

Aşa cum s-a menţionat, unul din elementele componente ale hamiltonianului este integrala de acoperire, definită prin relaţia:

$$S_{ij} = \int \varphi_i \, \varphi_j \, dv. \tag{3.1.75}$$

Acest capitol se va referi la cîteva aspecte legate de modul de calcul al unor integrale de acoperire (sau bicentrice).

Discuţia se va limita numai la elemente din perioada a II-a, ceea ce înseamnă că exemplificările se vor face folosind numai orbitali atomici $1s$, $2s$, $2p_x$, $2p_y$ şi $2p_z$. Folosind terminologia de legături σ şi π, cu ajutorul orbitalilor atomici menţionaţi se pot realiza combinaţiile din fig. 3.8.

Pentru a preciza definiţiile de legătură σ şi π trebuie subliniat că această terminologie se referă numai la o legătură bicentrică. În cazul legăturii σ, orbitalul molecular posedă o axă de simetrie care trece prin cele două nuclee, în timp ce legătura π este antisimetrică în raport cu aceeaşi axă.

În cazul în care doi orbitali p nu sînt paraleli, integrala de acoperire se calculează după o metodă care aminteşte de aspecte din algebra vectorilor. Fiecare orbital p se descompune în două componente, una perpendiculară pe axa care trece prin cele două nuclee, şi o altă componentă de-a lungul axei. Întrepătrunderea componentelor perpendiculare este tratată ca o acoperire π, în timp ce interpătrunderea componentelor de-a lungul axei are toate proprietăţile acoperirii σ (fig. 3.9), adică:

$$S_{tot} = \cos\alpha \cdot \cos\beta \cdot S(p_a, p_b, \sigma) + \sin\alpha \cdot \sin\beta \cdot S(p_a, p_b\, \pi) \tag{3.1.76}$$

unde S_{tot} este acoperirea totală, $S(p_a, p_b, \sigma)$ şi $S(p_a, p_b, \pi)$ sînt acoperirile calculate în ipoteza că orbitalii p_a şi p_b se acoperă

Scanned with OKEN Scanner

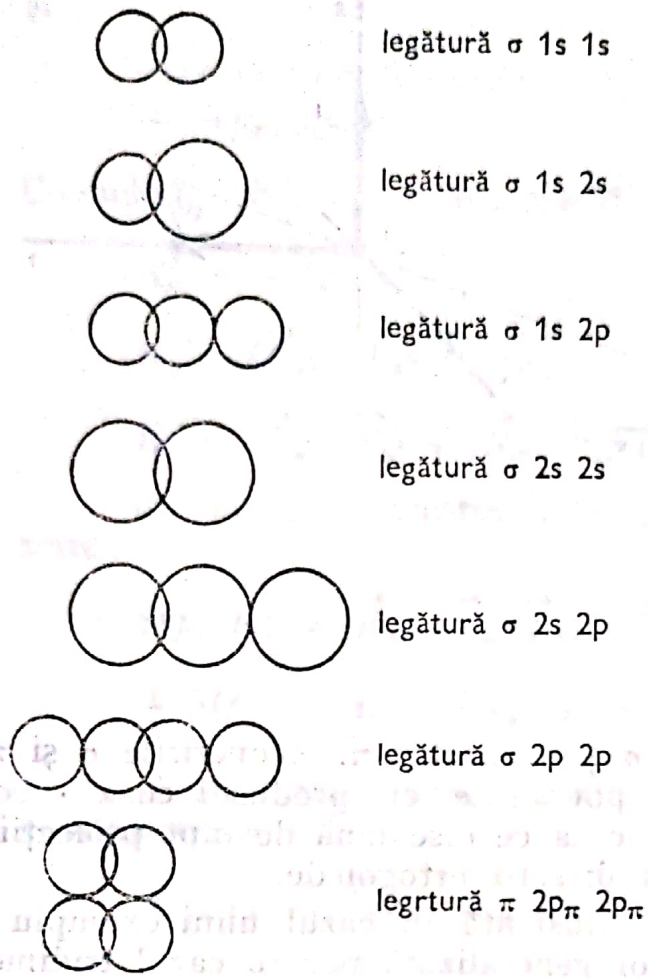

legătură σ 1s 1s
legătură σ 1s 2s
legătură σ 1s 2p
legătură σ 2s 2s
legătură σ 2s 2p
legătură σ 2p 2p
legrtură π 2pπ 2pπ

Fig. 3.8

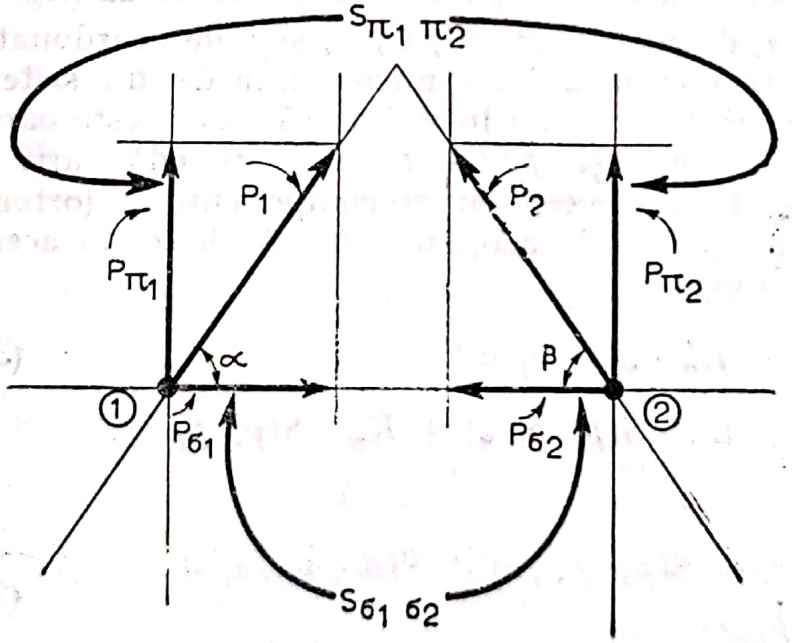

Sπ1 π2
P1
P2
Pπ1
Pπ2
α
β
①
②
Pб1
Pб2
Sб1 б2

Fig. 3.9

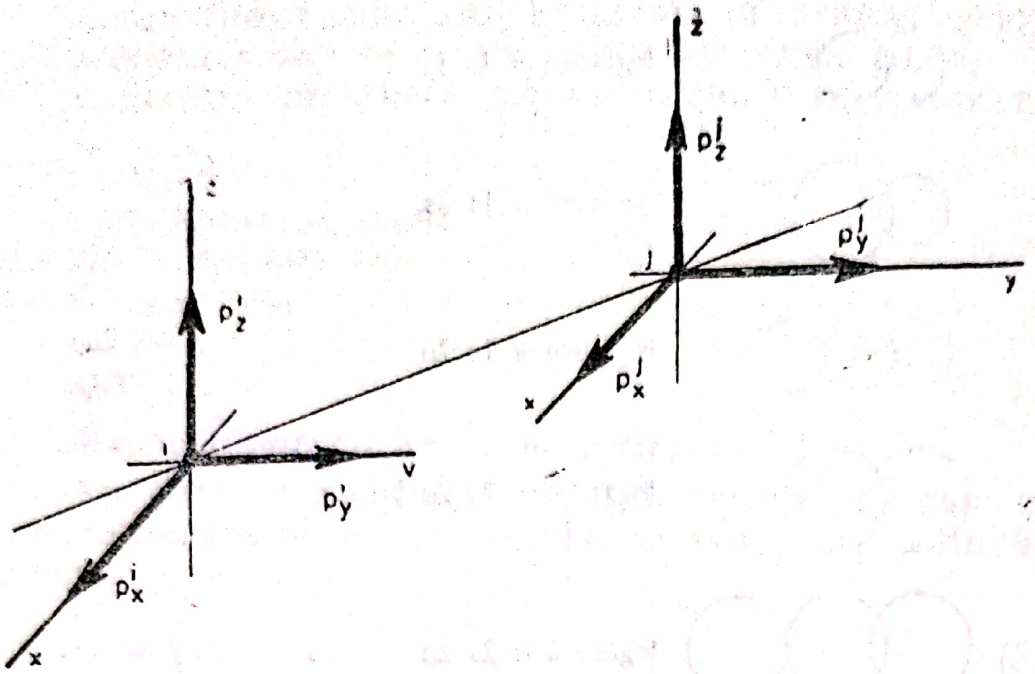


Fig. 3.10

conform unei legături σ şi respectiv π. Acoperirile σ şi π din formula (3.1.76) sînt ponderate cu produsul cosα · cosβ şi respectiv sin α · sin β, ceea ce înseamnă de fapt proiecţiile lui p_a şi p_b pe cele două direcţii ortogonale.

Această imagine, ilustrată în cazul unui exemplu bidimensional, poate fi uşor generalizată pentru cazul tridimensional, avîndu-se în vedere că se pot alege trei orbitali ortogonali, p_x, p_y şi p_z, fiecare definit în coordonatele locale, în aşa fel încît axele de coordonate să fie paralele două cîte două (fig. 3.10).

Punctele i, de coordonate x', y', z', şi j, de coordonate x'', y'', z'', reprezintă poziţia a două nuclee faţă de un sistem de coordonate ales arbitar. Avînd în vedere că axa ij este oarecare, orbitalii $p_i(p_{x'}, p_{y'}, p_{z'})$ şi $p_j(p_{x''}, p_{y''}, p_{z''})$ posedă atît componente σ (de-a lungul axei) cît şi componente π (ortogonale faţă de axă). Integralele de acoperire se calculează în acest caz astfel[28] (v. şi figurile):

$$S(s_i, p_j) = K'_\sigma \cdot S(s_i, p_j, \sigma) \tag{3.1.77}$$

$$S(p_i, p_j) = K_\sigma \cdot S(p_i, p_j, \sigma) + K_\pi \cdot S(p_i, p_j, \pi), \tag{3.1.78}$$

unde:

$$S(p_i, p_j, \sigma) = S(p_{x'}, p_{x''}, \sigma) + S(p_{y'}, p_{y''}, \sigma) + S(P_{z'} p_{z''}, \sigma) \tag{3.1.79}$$

Scanned with OKEN Scanner

şi

$$S(p_i, p_j, \pi) = (p_{x'}, p_{x''}, \pi) + S(p_{y'}, p_{y''}, \pi) + \\ + S(p_{z'}, p_{z''}, \pi) \tag{3.1.80}$$

Coeficienţii K'_σ, K_σ şi K_π rezultă din:

$$K'_\sigma = -(\vec{R}_{ab} \cdot \vec{P}_b) \tag{3.1.81}$$

$$K_\sigma = (\vec{R}_{ab} \cdot \vec{P}_a)(\vec{R}_{ba}, \vec{P}_b) \tag{3.1.82}$$

$$K_\pi = (\vec{P}_a \cdot \vec{P}_b) + (\vec{R}_{ab}, \vec{P}_a)(\vec{R}_{ba} \cdot \vec{P}_b) \tag{3.1.83}$$

Cu ajutorul acestor relaţii, integrala (3.1.78) se poate scrie:

$$S(p_i, p_j) = (\vec{R}_{ab} \cdot \vec{P}_a)(\vec{R}_{ba} \cdot \vec{P}_b)[S(p_a, p_b, \sigma) + \\ + S(p_a, p_b, \pi) + S(p_a \cdot p_b, \pi) \cdot (\vec{P}_a \cdot \vec{P}_b)] \tag{3.1.84}$$

$\vec{R}_{ij}$ reprezintă vectorul unitate de-a lungul axei ij şi are componentele:

$$\frac{x_i - x_j}{D}, \quad \frac{y_j - y_i}{D}, \quad \frac{z_j - z_i}{D}, \tag{3.1.85}$$

unde

$$D = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2} \tag{3.1.86}$$

Orbitalii atomici ai nucleului i şi ai nucleului j au următorii vectori unitari: $P_x(1,0,0)$, $P_y(0,1,0)$ şi $P_z(0,0,1)$.

Prin urmare, coeficienţii au următoarele valori; pentru integrale tip:

	$P_x - P_x$	$P_y - P_y$	$P_z - P_z$
K'	$-\frac{x_j - x_i}{D}$	$-\frac{y_j - y_i}{D}$	$-\frac{z_j - z_i}{D}$
K_σ	$-\left(\frac{x_j - x_i}{D}\right)^2$	$-\left(\frac{y_j - y_i}{D}\right)^2$	$-\left(\frac{z_j - z_i}{D}\right)^2$
K_π	$1 - \left(\frac{x_j - x_i}{D}\right)^2$	$1 - \left(\frac{x_j - x_i}{D}\right)^2$	$1 - \left(\frac{x_j - x_i}{D}\right)^2$

Scanned with OKEN Scanner

şi:

	$P_x - P_y$	$P_x - P_z$	$P_y - P_z$
K'	—	—	—
K_σ	$-\frac{(x_j - x_i)(y_j - y_i)}{D^2}$	$-\frac{(x_j - x_i)(z_j - z_i)}{D^2}$	$-\frac{(y_j - y_i)(z_j - z_i)}{D^2}$
K_π	$-\frac{(x_j - x_i)(y_j - y_i)}{D^2}$	$-\frac{(x_j - x_i)(z_j - z_i)}{D^2}$	$-\frac{(y_j - y_i)(z_j - z_i)}{D^2}$

Formulele cu ajutorul cărora se calculează integrala de acoperire depind de modul de alegere a funcţiilor care reprezintă orbitalii atomici. Selectarea tipului de orbitali se face avîndu-se în vedere şi dificultăţile care apar la calculul integralelor care intervin în teoria orbitalilor moleculari. Aşa-zisele funcţii hidrogenoide, schematizate de obicei în toate textele de chimie generală, anorganică sau organică, au partea radială sub formă de polinom, ceea ce complică extraordinar calculul integralelor (atît a acoperirii cît şi a celor de repulsie electronică). Prezintă un oarecare supliment de avantaje (fără a fi prea numeroase) funcţiile Slater[8], a căror parte radială are forma analitică:

$$R_{nl}(r) = (2\zeta)^{n+1/2}\,[(2n)!]^{-1/2} r^{n-1} \exp(-\zeta r) \qquad (3.1.87)$$

avînd

$$\zeta = \frac{Z - S}{n^*}, \qquad (3.1.88)$$

unde s — constanta de ecranare şi n^* — numărul cuantic principal efectiv sînt mărimi alese astfel ca să determine valori bune pentru nivelele energetice ale atomilor, raze atomice şi ionice corecte etc. De remarcat că pentru perioadele I şi II, în limitele căreia se încadrează obiectul acestui capitol, numărul cuantic principal n este egal cu numărul cuantic principal efectiv n^*. Valorile recomandate pentru sarcina efectivă $(Z-s)$ pentru elementele perioadelor I şi II sînt menţionate în tabelul 3.6.

Deşi orbitalii Slater sînt întrebuinţaţi în mod frecvent în calculele de orbitali moleculari, astăzi — odată cu dezvoltarea maşinilor de calcul — încep să fie întrebuinţate din ce în ce mai

Scanned with OKEN Scanner

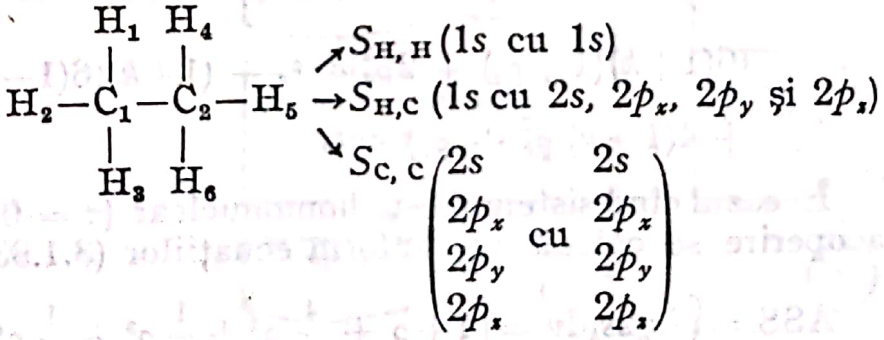
TABELUL 3.6.

VALOAREA SARCINII EFECTIVE IN ORBITALII ATOMICI SLATER

Z	Atom	Sarcina nucleară efectivă	
		1s	2s, 2p
1	H	1,00	
2	He	1,70	
3	Li	2,70	1,30
4	Be	3,70	1,95
5	B	4,70	2,60
6	C	5,70	3,25
7	N	6,70	3,90
8	O	7,70	4,55
9	F	8,70	5,20
10	Ne	9,70	5,85

mult funcţiile Gauss[9]. În aceste funcţii, partea radială are la exponentul părţii exponenţiale r^2, ceea ce creează numeroase avantaje la integrare. Volumul calculelor, şi nu dificultăţile de calcul, sînt însă mari, din această cauză întrebuinţarea funcţiilor Gauss în programe de calcul a structurii electronice pretinde calculatoare puternice.

Programul de calcul al integralelor de acoperire care urmează a fi prezentat foloseşte pentru orbitalii atomici $1s$, $2s$, $2p_x$, $2p_y$ şi $2p_z$ funcţii Slater. Pentru uşurarea urmăririi formulelor care vor fi utilizate, construirea programului, deşi sub o formă generală, va fi exemplificată în cazul concret al etanului.



O analiză succintă va permite imediat o generalizare. Există, aşa dar, 3 clase de integrale de acoperire, prima se referă la acoperirea orbitalilor $1s$ cu $1s$ (adică între atomii de hidrogen) $S_{H,H}$, a doua $1s$ cu orbitali $2s$ şi $2p$, adică între atomii de hidrogen şi carbon, $S_{H,C}$ şi în fine acoperirea orbitalilor atomici ai celor 2 atomi de carbon. În formulele de calcul sînt necesare pe lîngă

notaţia (3.1.88) şi următoarele (pentru notaţii vezi pag. 159 indicii a şi b se referă la nucleele a şi b):

$$\tau = \frac{\zeta_a - \zeta_b}{\zeta_a - \zeta_b} \ (a); \quad \rho = \frac{1}{2}(\zeta_a + \zeta_b) \cdot D \quad (b) \qquad (3.1.89)$$

$$k = \frac{1}{2}\left(\tau + \frac{1}{\tau}\right) \ (e); \ \rho_a = \zeta_a \cdot D \quad (c); \ \rho_b = \zeta_b \cdot D \quad (d)$$

$S_{H,H}$ se calculează cu ajutorul următoarei formule:

$$S_{H,H} = \int 1S_a \cdot 1S_a \, dv = \left(1 + \rho + \frac{1}{3}\rho^2\right)e^{-\rho} \qquad (3.1.90)$$

Orbitalii atomici ai hidrogenului se acoperă cu orbitalii atomici $2s$ şi $2p$ aparţinînd atomului de carbon; notăm în acest caz existenţa a două constante de ecranare Slater, corespunzătoare fiecărui atom de hidrogen şi respectiv de carbon. Formulele de calcul sînt:

$$S_{H,C} = \int 1s_A 1s_B dV = [(1-\tau^2)^{1/2}/\sqrt{3}\tau\rho]\,[-(1-k)\{2(1+k)(2-3k) + (1-2k)\rho_a\}e^{-\rho_a} + (1+k)\{2(1-k)(2-3k) + 4(1-k)\rho_b + \rho_b^2\}e^{-\rho_b} \qquad (3.1.91)$$

şi / sau

$$S_{H,C} = \int 1s_A 1s_B dV = [(1+\tau)/(1-\tau)^{1/2}(1/\tau\rho^2)]\,[-(1-k)^2\{6(1+k)(1+\rho_a) + 2\rho_a^2\}e^{-\rho_a} + (1+k)\{6(1-k)^2(1+\rho_a) + 4(1-k)\rho_b^2 + \rho_b^3\}e^{-\rho_b} \qquad (3.1.92)$$

În cazul cînd sistemul este homonuclear ($\tau = 0$), integralele de acoperire se calculează conform ecuaţiilor (3.1.93) — (3.1.96)

$$ASS = \int 2s_a 2s_b dv = \left[1 + \rho + \frac{4}{9}\rho^2 + \frac{1}{9}\rho^3 + \frac{1}{45}\rho^4\right]e^{-\rho} \qquad (3.1.93)$$

$$ASP = \int 2s_a 2p\sigma_b dv = \frac{1}{2\sqrt{3}}\rho\left[1 + \rho + \left(\frac{7}{15}\right)\rho^2 + \frac{2}{15}\rho^3\right]e^{-\rho} \qquad (3.1.94)$$

Scanned with OKEN Scanner

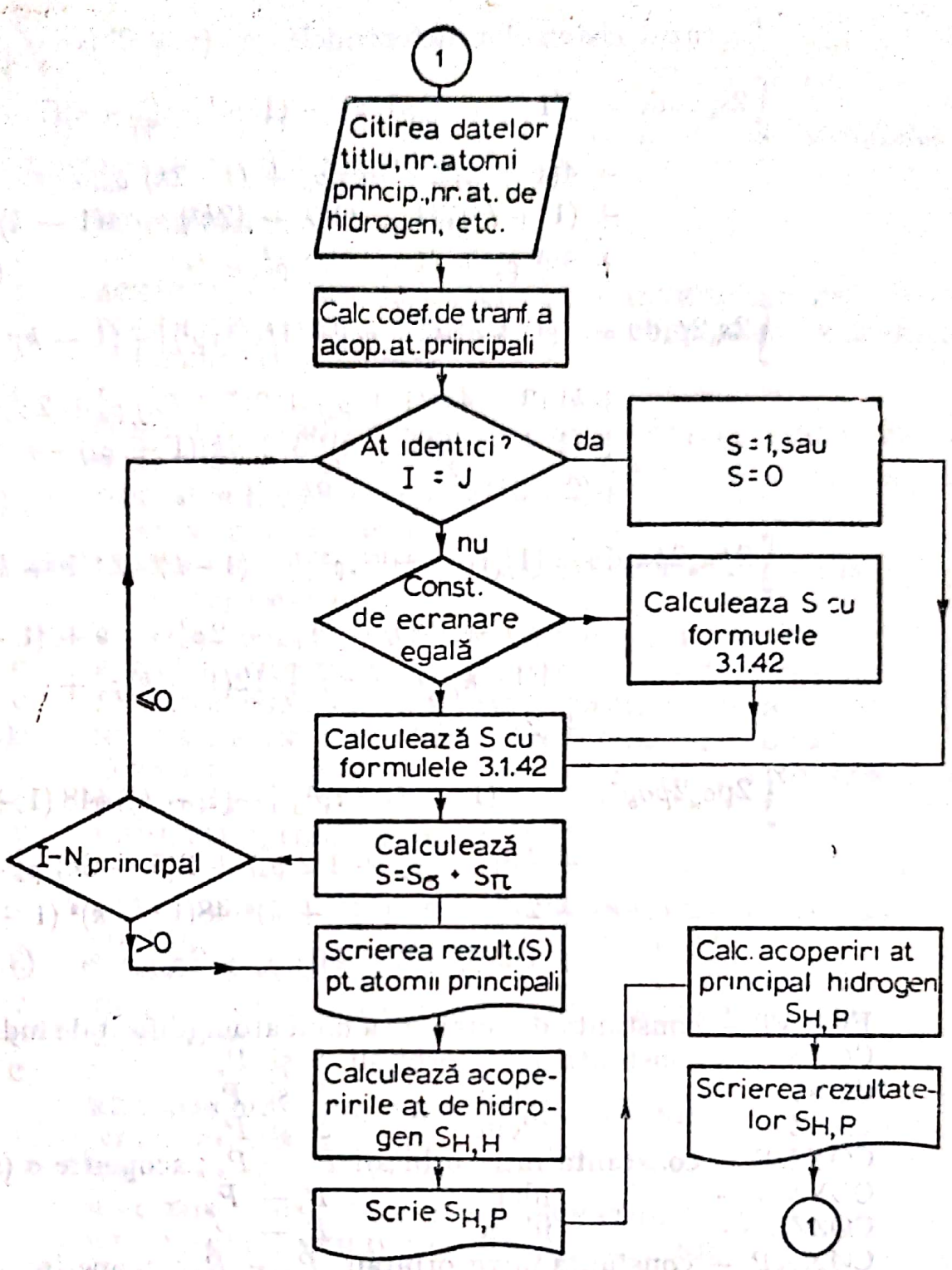


Fig. 3.11

$$APPP = \int 2p\Pi_a \cdot 2p\Pi_b dv = \left[1 + \rho + \frac{2}{5}\rho^2 + \left(\frac{1}{15}\right)\rho^3\right] e^{-\rho} \quad (3.1.95)$$

$$APPS = \int 2p\sigma_a \cdot 2p\sigma_b dv = \left[-1 - \rho - \frac{1}{5}\rho^2 + \frac{2}{15}\rho^3 + \frac{1}{15}\rho^4\right] e^{-\rho} \quad (3.1.96)$$

Scanned with OKEN Scanner

În cazul sistemelor heteronucleare ($\tau \neq 0$):

$$\int 2s_a 2s_b dv = [(1-\tau^2)^{1/2}/3\tau\rho][-(1-k)\{2(1+k)(7-12k^2)+ + 4(1+k)(2-3k)\,\rho_a + (1-2k)\,\rho_a^2\}\,e^{-\rho_a} + + (1+k)\{2(1-k)(7-12k^2) + 4(1-k)\,(2+ + 3k)\,\rho_b + (1+2k)\,\rho_b^2\}\,e^{-\rho_b} \quad (3.1.97)$$

$$\int 2s_a 2p_b dv = [(1+\tau)/(1-\tau)]^{1/2}(1/\sqrt{3}\tau\rho^2)[+(1-k)^2\{6(1+ + k)\,(3+4k)(1+\rho_a) + 2(5+6k)\,\rho_a^2 + 2\rho_a^3\}\,e^{-\rho_a} + + (1+k)\,\{6(1-k)^2(3+4k)(1+\rho_b) + 4(1-k) + + (2+3k)\,\rho_b^2 + (1+2k)\,\rho_b^3\}\,e^{-\rho_b} \quad (3.1.98)$$

$$\int 2p\pi_a 2p\pi_b dv = [1/(1-\tau^2)^{1/2}\tau\rho^3][-(1-k)^2\,\{24(1+k)^2(1+ + \rho_a) + 12(1+k)\,\rho_a^2 + 2\rho_a^3\}\,e^{-\rho_a} + (1+k)^2 \{24(1-k)^2(1+\rho_b) + 12(1-k)\,\rho_b^2 + + 2\rho_b^3\}\,e^{-\rho_b} \quad (3.1.99)$$

$$\int 2p\sigma_a 2p\sigma_b dv = [1/(1-\tau^2)^{1/2}\,\tau\rho^3]\,[-(1-k)^2\,\{48\,(1+ + k)^2(1+\rho_a + 1/2\,\rho_a^2) + 2(5+6k)\,\rho_a^3 + + 2\rho_a^4\}\,e^{-\rho_a} + (1+k)^2[48(1-k)^2\,(1+\rho_b + + 1/2\rho_b^2) + 2(5-6k)\rho_b^3 + 2\rho_b^4\}\,e^{-\rho_b} \quad (3.1.100)$$

ECRAT — constanta de ecranare a unui atom (diferit de hidrogen)
CØSX — constanta între orbitali S și P_x
CØSY idem S și P_y
CØSZ idem S și P_z
CØXXS — constanta între orbitali $P_x - P_x$; acoperire σ (sigma)
CØYYS — idem $P_y - P_y$
CØZZS — idem $P_z - P_z$
CØXXP — constanta între orbitali $P_x - P_x$; acoperire π (pi)
CØYYP — idem $P_y - P_y$; acoperire π
CØZZP idem $P_z - P_z$; acoperire π
CØXYS idem $P_x - P_y$ acoperire σ
CØXZS idem $P_x - P_z$ acoperire σ
CØYZS idem $P_y - P_z$ acoperire σ
CØXYP idem $P_x - P_y$ acoperire π

Scanned with OKEN Scanner

CØXZP	idem	$P_x - P_z$ acoperire π
CØYZP	idem	$P_y - P_z$ acoperire π

TAU, RHØ, RHØA, RHØB, KAP sînt denumirile variabilelor cu același nume din formulele 3.1.89.

```
      SUBRØUTINE ACØP(IN, IØ)
C
C     ACEASTA SUBRUTINA CALCULEAZA INTEGRALE DE ACØPE-
C     RIRE PENTRU UN SISTEM CARE CØNTINE NUMAI ELEMENTE
C     ALE PERIØADEI A DØUA
C
      CØMMØN/ INTEG/ X(10), Y(10), Z(10), XH(10), YH(10), ZH(10),
     1ECRAT(10), ECRAH, NØRB, S(40,40), G(40,40)
      DIMENSIØN TITLU(15)
      REAL KAP, KAPS, KAPD
 1000 READ(IN,100) NØCAZ, N, NH, ECRAH, (TITLU(I), I = 1,15)
      IF(NØCAZ) 1002,1002, 1001
 1001 READ(IN, 101) (X(I), Y(I), Z(I), ECRAT (I), I = 1, N)
C     NØCAZ REPREZINTA NUMARUL CAZULUI DE STUDIAT
C     NN ESTE EGAL CU NUMARUL ATØMILØR PRINCIPALI
C     NH REPREZINTĂ NUMARUL ATØMILOR DE HIDRØGEN
C     ECRAH ESTE CØNSTANTA DE ECRANARE A ATØMILOR  DE
C     HIDRØGEN
C     URMEAZA TITLUL PRØBLEMEI
C
C     CARTELELE URMATØARE CØNTIN INFORMATII DESPRE
C     CØØRDØNATELE X, Y, Z ALE ATØMULUI ,,I''
C     PØSEDIND CØNSTANTA DE ECRANIZARE ECRAT (I)
C     IN FINE, URMEAZA CØØRDØNATELE ATØMILØR  DE
C     HIDRØGEN
C
      READ(IN, 102) (XH(I), YH(I), ZH(I), I = 1, NH)
      WRITE(IØ, 103) (TITLU(I), I = 1, 15)
      WRITE(IØ, 104) NØCAZ, N, NH
      WRITE(IØ, 101) (X(I), Y(I), Z(I), ECRAT(I), I = 1,N)
      WRITE(IØ, 102) (XH(I), YH(I), ZH(I), I = 1, NH)
      WRITE(IØ, 105)
      C15 = 1./15.
      C9 = 1./9.
      C45 = 1./45.
      CPI = 1./3.4641
      DØ 1 I = 1,  N
      IX = I + N
```


Scanned with OKEN Scanner

```
      IY = IX + N
      IZ = IY + N
      DØ 2 J = 1, N
      JX = J + N
      JY = JX + N
      JZ = JY + N
C
C     SE VERIFICA DACA CEI DØI ØRBITALI APARTIN ACELUIASI
C     ATØM
C
      IF(I-J) 10,11,10
C
C     CALCULUI DISTANTEI INTERATØMICE SI A
C     PARAMETRILØR DIRECTIØNALI AI ØRBITALILØR ATØMILOR
C     PRINCIPALI APARTININD PERIØADEI A DØUA
C
   10 D = SQRT ((X(J)-X(I))**2 + (Y(J)-Y(I))**2 +
     1 (Z(J)-Z(I))**2)
      CØSX = (X(J)-X(I))/D
      CØSY = (Y(J)-Y(I))/D
      CØSZ = (Z(J)-Z(I))/D
      CØXXS = CØSX**2
      CØYYS = CØSY**2
      CØZZS = CØSZ**2
      CØXYS = CØSX*CØSY
      CØXZS = CØSX*CØSX
      CØYZS = CØSY*CØSZ
C
C     CARE ESTE APARTENENTA CELØR DØI ØRBITALI ?
C
      IF (ECRAT(I)-ECRAT(J)) 13,12,13
   12 RHØ = ECRAT(I)*D/.5292
      EP = EXP(-RHØ)
      RHØ2 = RH**2
      ASS = (1. + RHØ + 4.*C9*RHØ2 + C9*RHØ2*RHØ+C45*RHØ2**2)
     1 *EP
      ASP = EP*(-1.-RHØ-.2*RHØ2+2.*C15*RHØ2*RHØ+C15*
     1 RHØ2*RHØ)
      APPP = EP*(1. + RHØ+.4*RHØ2 + C15*RHØ2*RHØ)
      GØ TØ 20
   13 RHØA = ECRAT(I)*D/.5292
      RHØB = ECRAT (J)*D/.5292
      RHØ = .5*(RHØA + RHØB)
```


Scanned with OKEN Scanner

```
      TAU = (RHØA−RHØB)/(RHØA + RHØB)
      KAP = .5*(TAU + 1./TAU)
      KAPS = 1. + KAP
      KAPD = 1.−KAP
      RHØA1 = 1. + RHØA
      RHØB1 = 1. + RHØB
      TSQ = SQRT (1.−TAU**2)
      TT = SQRT ((1. + TAU) (1.−TAU)*. 5773503/(TAU*RHØ*2.)
      QQ = 14.−24.*KAP*KAP
      QQQ = 18. + 24.*KAP
      RHØPA = 48.*(1. + RHØA + .5*RHØA*RHØA)
      RHØPB = 48.*(1. + RHØB + .5*RHØB*RHØB)
      EPA = EXP(−RHØA)
      EPB = EXP(−RHØB)
      ASS = TSQ/(3.*TAU*RHØ) (−EPA*KAPD*(KAPS*QQ + 4.
     1 * KAPS*(2.−3.*KAP)*RHØA + (1.−2.*KAP)* RHØA *RHØA)+
     2 EPB*KAPS*KA PD*QQ + 4.*KAPD* (2. + 3.*KAP)*RHØB+
     3 (1. + 2.*KAP)*RHØB*RHØB))
        ASP = TT*(−EPA*KAPD*KAPD*(QQQ*KAPS*RHØA1 + 2.*
     1 (5. + 6.*KAP)*RHØA*RHØA + 2.*RHØA**3) + EPB*KAPS*
     2 (QQQ*KAPD*KAPD *RHØB1 + 4.*KAPD*(2. + 3.*KAP)*RHØB*
     3 RHØB + (1. + 2.*KAP)*RHØB**3))
        APPS = 1./(TSQ*TAU*RHØ**3)*(−EPA*KAPD* KAPD*
     1 (KAPS*KAPS*RHØPA + 2.*(5. + 6.*KAP)*RHØA**3 + 2.*
     2 RHØA**4) + EPB*KAPS*KAPS*(KAPD*KAPD*RHØPB + 2.* (5.
     3 −6.*KAP)*RHØB**3 + 2.* RHØB**4))
        APPP = 1./(TSQ*TAU*RHØ**3)*(−EPA*KAPD*KAPD*
     1 (24.*KAPS*KAPS*RHØA1 + 12.*KAPS*RHØA*RHØA + 2.*
     2 RHOA**3) + EPB*KAPS*KAPS*(24.*KAPD*KAPD + RHØB1
     3 +12.*KAPD*RHØB*RHØB + 2.* RHØB**3)
      GØ TØ 20
      D = 0.
      ASS = 1.
      ASP = 0.
      APPS = 1.
      APPP1 = 1.
      S(I,J) = 1.
      S(I,IX) = 0.
      S(I,IY) = 0.
      S(I,IZ) = 0.
      S(IX,IY) = 0.
      S(IX,IZ) = 0.
```


Scanned with OKEN Scanner

```
      S(IY,IZ) = 0.
      GØ TØ 21
   20 APPT = APPS + APPP
      S(I,J) = ASS
      S(I,JX) = - CØSX*ASP
      S(I,JY) = -CØSY*ASP
      S(I,JZ) = - CØSZ*ASP
      S(IX,JX) = - CØXXS*APPT + APPP
      S(IY,JY) = - CØYYS*APPT + APPP
      S(IZ,JZ) = - CØZZS*APPT + APPP
      S(IX,JY) = - CØXYS*APPT
      S(IX,JZ) = - CØXZS*APPT
      S(IY,JZ) = - CØYZS*APPT
   21 WRITE(IØ,106) I,J, ECRAT(I), ECRAT(J), D, ASS,
     1 ASP, APPS, APPP
    2 CØNTINUE
    1 CØNTINUE
      N4 = N*A
      N5 = N4 + NH
      N51 = N5 + 1
      N41 = N4 + 1
      DØ 3 J = 1,I
    3 S(N51-I,N51-J) = S(N41-I,N41-J)
      WRITE (IØ,107)
C
C     CALCULUL ACØPERIRII ATØMILØR DE HIDRØGEN.
C
      DØ 4 J = 1,NH
      DØ 4 I = 1, NH
      D = SQRT((XH(J) - XH(I))**2 + (YH(J) - YH(I))**2 +
     1          +(ZH(J) - ZH(I))**2)
      RHØH = ECRAH*D/.5292
      S(I,J) = (EXP(-RHØH))*(1. + RHØH + RHØH*RHØH/3.)
    4 WRITE(IØ,108) I,J,D,S(I,J)
      WRITE(IØ,109)
C
C     URMEAZA CALCULUL ACØPERIRILOR ATØM PRINCIPAL -
C     HIDRØGEN
C
      DØ 6 J = 1,N
      DØ 6 I = 1,NH
      D = SQRT((X(J)-KH(I))**2 + (Y(J)-YH(I))**2 + (Z
     1(J) - Z(I))**2)
```


Scanned with OKEN Scanner

```
      CØSX = (X(J)−XH(I))/D
      CØSY = (Y(J) − YH(I))/D
      CØSZ = (Z(J)−ZH(I)I/D
      RHØA = ECRAH*D/.5292
      RHØB = ECRAT(J)*D/.5292
      RHØ = .5*(RHØA + RHØB)
      TAU = (RHØA−RHØB)/(RHØA + RHØB)
      KAP = .5*(TAU + 1./TAU)
      EPA = EXP(−RHØA)
      EPB = EXP(−RHØB)
      AHS = (SQRT((1.−TAU*TAU)/3.)/(TAU*RHØ))*(−EPA*(1.
     1 −KAP)*(2.*(1. + 4.*(1.−KAP)*(2.−3.*KAP) + RHØA*(1.−
     2 2.*KAP)) + EPB (1. + KAP)*(2.*(1.−KAP)*(2.−3.*KAP)
     3 + 4.*(1.−KAP)*RHØB + RHØB*RHØB))
      AHP = (SQRT((1. + TAU)/(1.−TAU))*(1./(TAU*RHØ*
     1 RHØ))*(−EPA*(1.−KAP)**2*(6.*(1. + KAP)*(1.RHØA) + 2.
     2 *RHØA*RHØA) + EPB*(1. + KAP)*(6.*(1.−KAP)**2*(1. +
     3 RHØB) + 4.*(1.−KAP)*RHØB*RHØB* + RHØB**3))
      WRITE (IØ,110) J,I,D,AHS,AHP
      JS = J + NH
      JX = JS + N
      JY = JX + N
      JZ = JY + N
      S(I,JS) = AHS
      S(I,JX) =− CØSX*AHP
      S(I,JY) = − CØSY*AHP
    6 S(I,JZ) = − CØSZ*AHP
      DØ 7 I = 1,N5
      S(I,I) = 1.
      DØ 7 J = 1,I
    7 S(I,J) = S(J,I)
      WRITE(IØ,111)
      KITE = 0
   17 LØW = KITE + 1
      KITE = KITE + 14
      IF(KITE−N5) 14,14,15
   14 KITE = KITE
      GØ TØ 16
   15 KITE = N5
      GØ TØ 17
   16 WRITE (IØ,112) (I,(S(I,J), J = LØW,KITE)
      GØ TØ 17
   17 WRITE(IØ,112) (I,(S(I,J), J = LØW, KITE)
```


Scanned with OKEN Scanner

```
100 FØRMAT (3I2, F5.3,15A4)
101 FØRMAT  (4F10.7)
102 FØRMAT  (3F10.7)
103 FØRMAT (15A4)
104 FØRMAT ('CAZUL NØ. = ',I2,///,'NUMARUL TØTAL
   1 AL ATØMILØR PRINCIPALI = ',I2,///,'NUMARUL ATØMILØR
   2 DE HIDRØGEN = ', I2,///)
105 FØRMAT ('ATØM(I)'2X,'ATØM(J)',2X,'CØEFICIENTI
   1 SLATER', 2X,'DISTANTA', 2X, 'ACØP S—S', 2X, 'ACØP
   2 S—P', 2X,'ACØP P—P SIGMA', 2X, 'ACØP P—P PI',//)
106 FØRMAT (3X,I2,7X,I2,11X,F5.3,7X,F10.7, 2X,F7.5)
107 FØRMAT ('INTEGRALA DE ACØPERIRE A ATØMILOR DE
   1 HIDRØGEN')
108 FØRMAT (10X, I2,4X,I2, 4X, F7.5)
109 FØRMAT ('INTEGRALA DE ACØPERIRE A ATØMILØR
   1 PRINCIPALI CU ATØMII DE HIDRØGEN,(//,'ATØM
   2 PRINCIPAL', 2X, 'HIDRØGEN', 2X, 'CØMPØNENTA SIGMA' 2X,
   3 'CØMPØNENTA PI')
110 FØRMAT (///,4X,2I2, 13X, 2I2, 10X, F5.4, 9X, F7.5,///)
111 FØRMAT (30X,'MATRICEA INTEGRALELØR DE ACØPERIRE')
    RETURN
    END
```

Exemplu de utilizare a programului HESP (Metoda Hückel Extinsă pentru elemente care posedă orbitali atomici S şi P).

Modul de alcătuire a datelor de citire se ilustrează pe cele două conformaţii ale etanului: (a) conformaţia eclipsată şi (*b*) conformaţia intercalată. Considerînd legăturile C—C = 1,54Å şi C—H = 1,096Å, toate unghiurile de valenţe tetraedrice, originea axelor de coordonate la mijlocul legăturii C—C, avînd axa x de-a lungul acestei legături (v. fig.) $H_{ii}^{2s} = -21{,}4$eV, $H_{ii}^{2p} = -11{,}4$eV, constanta de ecranare (coeficientul Slater) al hidrogenului 1,0, în timp ce valoarea aceleiaşi variabile în cazul carbonului este 1,625; termenii nediagonali se calculează cu relaţia Wolfsberg-Helmholtz (DECI IND = 0), $K = 1{,}75$.

(a) conformaţia eclipsată (fiecare rînd reprezintă o cartelă):

```
1.75
0102061.000 ETAN CØNFØRMATIA INTERCALATA
  0.          0.77        0.          1.625
  0.         —0.77        0.          1.625
  1.037089    1.136666    0.
—0.51854      1.13666    —0.898146
—0.51854      1.13666     0.898146
```


Scanned with OKEN Scanner

```
  0.51854        −1.13666      0.898146
−1.03709         −1.13666      0.
  0.51854        −1.13666     −0.898146
−21.400−21.400
−11.400−11.400
  0
```

Atenţie: Ultima cartelă conţine 0. numai dacă se execută un singur caz; dacă există şi alte cazuri se continuă cu prima cartelă de la cazul următor.

(b) conformaţia eclipsată:

```
1.75
0202061.000 ETAN CØNFØRMATIE ECLIPSATA
  0.              0.77          0.          1.625
  0              −0.77          0.          1.625
  1.037089        1.13666       0.
−0.51854          1.13666      −0.898146
−0.51854          1.13666       0.898146
  1.037089       −1.13666       0.
−0.518544        −1.13666       0.898146
−0.51854         −1.13666      −0.898146
−21.400−21.400
−11.400−11.400
  0.
```

3.1.6. METODE AUTOCOERENTE DE CALCUL AL STRUCTURII ELECTRONICE A MOLECULELOR FOLOSIND TOŢI ELECTRONII DE VALENŢĂ

Recapitulînd istoria chimiei cuantice, istorie care începe în jurul anilor 1930, se constată că aceasta a evoluat în mod firesc de la metode simple, naive, la metode complexe, care necesită un număr extraordinar de operaţii. Evoluţia chimiei cuantice aplicate este strîns legată de evoluţia mijloacelor pe care aceasta le utilizează, adică a chimiei cuantice teoretice pe de o parte, iar pe de altă parte a mijloacelor practice de realizare a calculelor, adică a calculatoarelor, în special din punctul de vedere al creşterii memoriei interne.

Sistemele electronice π au constituit primele modele pe care s-au aplicat metodele elaborate de chmia cuantică teoretică. Chimiştii au realizat că restricţia impusă de condiţia de planaritate a unei molecule pentru a face posibilă aplicarea

Scanned with OKEN Scanner

metodelor Hückel şi PPP este foarte limitativă. Această condiţie nu este îndeplinită decît de un număr foarte restrîns de molecule, or interesul chimiştilor depăşeşte cu mult această sferă; moleculele care posedă schelete σ şi π, sau numai schelete σ, întrec cu mult numărul sistemelor π. Prima tentativă de calcul, în care s-au folosit toţi electronii de valenţă ai atomilor care alcătuiesc o moleculă, a fost aceea a lui Wolfsberg şi Helmholtz[19] în 1952, cînd s-a calculat structura electronică a anionilor MnO_4^-, CrO_4^{2-} şi ClO_4^-. Aspecte ale metodei, ca de exemplu alegerea elementului de matrice H_{ii} egal cu potenţialul de ionizare a stării de valenţă respective, în timp ce elementul nediagonal H_{ij} se considera ca o medie aritmetică a termenilor diagonali şi ponderaţi cu integrala de acoperire S_{ij} (v. pag. 190), au constituit de asemenea aproximaţiile pe care s-a elaborat metoda Hoffmann (v. pag. 183). Aceste procedee reprezintă însă aproximaţii severe; un progres deosebit s-a realizat abia în jurul anului 1965 cînd s-au pus bazele metodologiei unor metode bazate pe cîmp autocoerent (v. pag. 133), introducîndu-se şi o parte din integralele de repulsie electronică, considerate a fi cele mai importante.

Variantele de rezolvare a ecuaţiilor lui Roothaan folosind toţi electronii de valenţă s-au dezvoltat rapid constituind o adevărată explozie de metode de calcul. Desigur nu toate s-au bucurat de succes. Unele au fost uitate, deşi ar putea conţine aproximaţii interesante, altele, puţine la număr, domină cu autoritate; un număr impresionant de calcule s-au efectuat şi se efectuează cu metode ca *CNDØ, INDØ**. Există mai multe motive care explică succesul acestor metode. În primul rînd este vorba de existenţa unei justificări teoretice, adică ecuaţiile utilizate pentru construcţia matricii Fock (hamiltonianul sistemului) sînt demonstrabile. În al doilea rînd, aceste metode apelează la aproximaţii foarte plauzibile, fiind cele mai recomandate din mulţimea aproximaţiilor existente la această dată. În al treilea rînd poate la fel de important ca celelalte două pentru vitalizare şi vitalitatea unor metode este faptul că aceste metode sînt public accesibile, fie prin QCPE (v. pag. 125), fie au fost publicate în Apendix la diferite cărţi cum ar fi referinţa 6.

Aşa cum am menţionat în procedeul de calcul al sistemelor π, o contribuţie majoră a fost adusă de J.A. Pople, contribuţie care practic a dominat chimia cuantică aplicată în perioada 1952—1964. Este acelaşi nume care a introdus după

---

* prescurtările utilizate sînt explicate la pag. 207 şi 208.

Scanned with OKEN Scanner

anul 1965, procedeul cel mai frecvent folosit în calculul cu toţi electronii de valenţă, rezumat în metodele *CNDØ* şi *INDØ*

Aspecte ale metodei CNDØ. Problema integralelor.

O inspecţie chiar şi sumară a ecuaţiilor lui Roothaan atrage atenţia că pentru rezolvarea acestor ecuaţii este necesar calculul unui număr impresionant de integrale. Desigur că nu toate au o însemnătate egală, dictată în primul rînd de mărimea lor. Unele au valori de ~ 10 eV, altele sînt subunitare. O parte din integrale se calculează relativ rapid, altele necesită însă o suită de calcule numeroase. În general, dacă sistemul conţine n orbitali atomici cu care se construieşte funcţia de undă moleculară, numărul de integrale de calculat este proporţional cu n^4. Benzenul are 6×4 orbitali atomici aparţinînd atomilor de C plus 6×1 aparţinînd atomilor de hidrogen, în total 30 de orbitali atomici, ceea ce face ca numărul de integrale de calculat să fie de ordinul $30^4 = 810.000$. Dar benzenul este unul din cele mai simple sisteme. Decalina este un caz în care s-ar cere calculate $(10 \times 4 + 18 \times 1)^4$ integrale. Este evident că apare ca obligatorie introducerea unor aproximaţii prin care să se reducă substanţial numărul integralelor necesare în calcul. Metodele menţionate mai sus îşi derivă prescurtarea din modul în care se reţin anumite integrale, mai exact din natura aproximaţiei generale care face ca unele integrale să devină automat zero. A. *CNDØ* — înseamnă *C*omplete *N*eglect of *D*ifferential *O*verlap, adică neglijarea totală a acoperirii diferenţiale (dS). Această mărime (dS) fiind (i şi j aparţin la două centre diferite)

$$dS = \varphi_i, \varphi_j \quad i \in A \; ; j \in B \tag{3.1.101}$$

înseamnă că în aproximaţia *CNDØ* ea se reduce la

$$dS = 0. \tag{3.1.102}$$

Există justificarea teoretică pentru această aproximaţie. Dacă se face o ortogonalizare a orbitalilor moleculari prin procedeul Löwdin (v. pag. 184) această condiţie ($dS = 0$) este îndeplinită în mod automat. Deci metoda *CNDØ* reclamă un segment de prelucrare a matricei Fock şi a vectorilor proprii, similar cu procedeul descris la problema de valori proprii cînd se utilizează în calculele şi integrala de acoperire (v. pag. 184).

Rezultatele acestei aproximaţii constau în neglijarea:

(a) Integralelor tricentrice

$(ij, k\,k)$, unde i şi j şi k sînt orbitali diferiţi

$$(ij, kk) = \int \Phi_i(1) \cdot \Phi_j(1) \frac{1}{r_{12}} \Phi_k(2)\Phi_k(2)dV$$

Scanned with OKEN Scanner

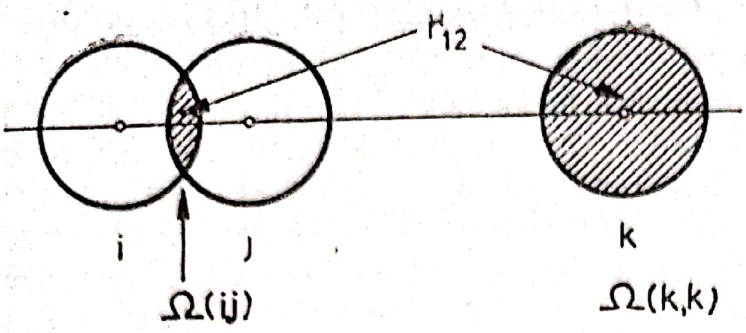


Fig. 3.12

(b) Integrale tetracentrice (fig. 3.13)

(*ij*, *kl*)

Părţile haşurate sînt distribuţiile electronice Ω între care apare repulsia.

Rezultatele calculelor trebuie să fie invariante faţă de sistemul de coordonate în care s-a definit geometria moleculei. Această cerinţă introduce a treia aproximaţie în calculul integralelor de repulsie electronică. În metoda *CNDØ*, integralele de repulsie electronică care se conformează condiţiei de invarianţă la sistemul de coordonate sînt numai acelea de forma:

$$\gamma_{AB} = (s_A s_A,\ s_B s_B) \tag{3.1.103}$$

unde *A* şi *B* se referă la atomii (şi nu orbitali atomici) între care se calculează integrala de repulsie electronică. Aşadar, şi repulsiile de tipul $(P_x^A\ P_x^A,\ P_x^B\ P_x^B)$ $(P_y^A\ P_y^A,\ P_y^B\ P_y^B)$ etc. se aproximează cu formule de tipul

$$\gamma_{AB} = (s_A s_A,\ s_B s_B)$$

B. În aproximaţia *INDØ* (*I*ntermediate *N*eglect of *D*ifferential *O*verlap) în cazul în care orbitalii diferiţi aparţin aceluiaş atom

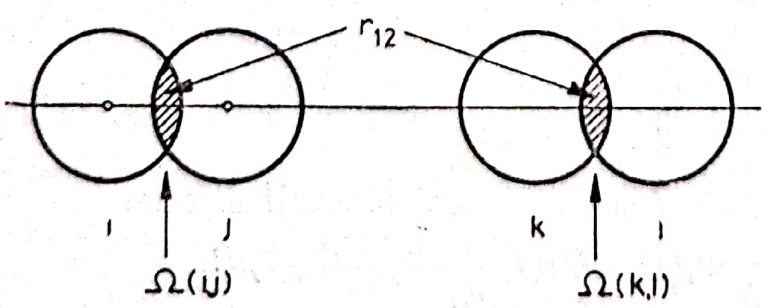


Fig. 3.13

Scanned with OKEN Scanner

(elementele perioadei a doua participă cu 4 orbitali atomici) astfel de integrale vor fi reţinute, adică

$(ij,kk) \neq 0$ dacă i, j, k aparţin atomului A (de ex. s, px, py)

$(ij, ik) \neq 0$ dacă i, j, k aparţin atomului A

$(ij, kl) \neq 0$ dacă i, j, k, l aparţin atomului A

Aceste integrale sînt egale cu parametrii Slater-Condon, F^0, G^1 şi F^2. Astfel[29]:

$$(ss, ss) = (ss, pp) = F^0 = \gamma_{AA}$$

$$(sp, sp) = \frac{1}{3} G^1$$

$$(pp', pp') = \frac{3}{25} F^2 \qquad (3.1.104)$$

$$(pp, pp) = F^0 + \frac{4}{25} F^2$$

$$(pp, pp') = F^0 - \frac{2}{25} F^2$$

Simbolizarea (pp', pp') reprezintă integrale de tipul $(P_x\ P_y, P_x, P_z)$. F^0 se alege ca fiind γ_{AA}, în timp ce F^2 şi G^1 reprezintă cele mai bune valori ale acestor mărimi, care reproduc date spectroscopice ale atomilor respectivi. Aceste mărimi sînt prezentate în Tabelul 3.7.

TABELUL 3.7.

VALORI EMPIRICE ($\times 10^3$) PENTRU G^1 ŞI F^2

Atom	Li	Be	B	C	N	O	F
G^1	9.2012	14.07	19.9265	26.7708	34.6029	43.423	53.2305
F^2	4.9865	8.9125	13.041	17.372	21.9055	26.6415	31.580

Justificarea formulelor de lucru, cu ajutorul cărora se calculează elementele de matrice, ar prezenta, desigur, un interes major. În felul acesta am cunoaşte exact care sînt aproximaţiile care stau la baza metodei, urmînd ca programatorul să acţioneze în viitor în funcţie de progresul realizat în domeniul respectiv şi asupra acestor puncte, poate paralel cu unele îmbunătăţiri ale modului de programare. Prezentarea aspectelor teoretice sînt

Scanned with OKEN Scanner

obiectul chimiei cuantice, depășind limitele acestui material, de aceea cititorul este rugat să accepte (ca fapt și în alte prezentări din cadrul acestei cărți) principalele formule fără a fi demonstrate.

Modul de calcul al elementelor de matrice derivă direct din teoria generală a lui Roothaan (v. 3.1.8.). În versiunea CNDØ-2 elementele de matrice sînt următoarele:

$$F_{\mu\mu} = H_{\mu\mu} + \left(P_{AA} - \frac{1}{2} P_{\mu\mu}\right)\gamma_{AA} + \sum_{B(\neq A)} P_{BB}\gamma_{AB}$$
$$F_{\mu\nu} = H_{\mu\nu} - \frac{1}{2} P_{\mu\nu} \cdot \gamma_{AB} \qquad (3.1.105)$$

în care:

$$H_{\mu\mu} = -1/2(I_\mu + A_\mu) - (Z_A - 1/2)\gamma_{AA} - \sum_{B(\neq A)} Z_B\gamma_{AB}$$
$$H_{\mu\nu} = 1/2(\beta_A^0 + \beta_B^0)S_{\mu\nu} \qquad (3.1.106)$$

avînd:

$$P_{\mu\nu} = 2\sum_{i}^{ocup} C_{i\mu} \cdot C_{i\nu} \qquad (3.1.107)$$

și densitatea electronică a atomului A

$$P_{AA} = \sum_{\mu}^{A} P_{\mu\mu} \qquad (3.1.108)$$

Sînt folosite următoarele notații:

— μ și ν se referă la orbitalii atomici, aparținînd atomilor A și respectiv B

— I_μ și A_μ sînt potențialele de ionizare și afinitatea electronică a electronului μ (v. Tabelul 3.9).

P_{AA} — sarcina electronică totală a atomului A

$P_{\mu\mu}$ — sarcina electronică a orbitalului μ, aparținînd atomului A

$P_{\mu\nu}$ — ordinul de legătură între orbitalii atomici μ (aparținînd atomului A) și ν (aparținînd atomului B)

β_{AB}^0 — parametru semiempiric de legătură între atomii A și B. Se calculează $\beta_{AB}^0 = (1/2)(\beta_A^0 + \beta_B^0)$ (v. Tabelul 3.8).

Z_A și Z_B — sarcina nucleară efectivă (egală cu Z — numărul electronilor de valență, de ex.: $Z_{C^{4+}} + = 8$, $Z_{N^{5+}} + = 10$

Scanned with OKEN Scanner

γ_{AA}, γ_{AB} — integrale de repulsie electronică (v. descrierea formulelor de calcul).

$S_{\mu\nu}$ — integrala de acoperire (v. capitolul 3.1.5).

CNDØ-2 este o metodă de calcul în cîmp autocoerent, elementele de matrice $F_{\mu\mu}$ şi $F_{\mu\nu}$ sînt funcţii de coeficienţii orbitalilor moleculari care sînt necunoscuţi. Întocmai ca la toate metodele de calcul prin tehnica cîmpului autocoerent, setul iniţial al acestor coeficienţi se calculează în aproximaţie

$$F_{\mu\mu} = H_{\mu\mu} \tag{3.1.109}$$

$$F_{\mu\nu} = H_{\mu\nu} \tag{3.1.110}$$

utilizînd pentru β_A^0 şi $1/2\ (I_\mu + A_\mu)$ valori indicate în Tabelul 3.8. şi Tabelul 3.9. (valorile se referă dacă se utilizează aproximaţiile metodei *CNDØ*-2).

TABELUL 3.8. **VALOAREA PARAMETRULUI β^0 (în eV)**

Atom	*H*	Li	Be	B	C	N	O	F
$-\beta_A^0$	9	9	13	17	21	25	31	39

TABELUL 3.9. **VALOAREA TERMENULUI $1/2\ (I\mu + A\mu)$ (în eV)**

Atom		H	Li	Be	B	C	N	O	F
$1/2(I_\mu+A_\mu)$	*S*	7.176	3.106	5.946	9.594	14,051	19.316	25.390	32.232
	P		2.563	2.563	4.001	4.472	7.275	9.111	11.080

Formule de calcul în aproximaţia zero sînt derivate din (3.1.105) avînd în vedere că în această etapă $P_{ij} = 0$.

Diagonalizarea acestei matrice cu metoda Jacobi sau Givens oferă automat primul set de coeficienţi cu care se recalculează elementele $F_{\mu\mu}$ şi $F_{\mu\nu}$ dar de data aceasta cu formulele (3.1.105). Calculul se repetă pînă cînd se satisface un criteriu de convergenţă (energie totală, energiile nivelelor moleculare, coeficienţii funcţiilor de undă).

Un program de calcul *CNDØ* poate fi structurat într-un program principal în care să se calculeze elementele de matrice Fock (*F*), şi mai multe subrutine, necesare calculului integra-

Scanned with OKEN Scanner

lelor de acoperire, integralelor de repulsie interelectronică, subrutină de diagonalizare a matricei Fock, subrutină de calcul a ordinelor de legătură. Din această înșiruire se poate remarca că astfel de exemple au fost deja formulate în materialul anterior.

Programul principal, a cărui organigramă este următoarea;

```
C     PRØGRAM DE CALCUL A STRUCTURII ELECTRØNICE
C     PRIN METØDA CNDØ-2
C
      DIMENSIØN BETAØ (40), ZN(40), SC(40,40), Q(40),
     1                  WIA (20)
      CØMMØN/INTEG/X(10), Y(10), Z(10), XH(10), YH(10),
     1                  ZH(10)ECRAT(10)
      ECRAH, NØRB, S(40,40), G(40,40)
      CØMMØN/EIG/H(40,40), F(40,40), HB(40,40)
      IN = 103
      IØ = 105
C
C     N REPREZINTA NØ. ATØMILØR PRINCIPALI DIN
C     MØLECULA NH PREPREZINTA NØ. ATØMILØR DE
C     HIDRØGENI
C     READ (IN,1000) N, NH, ECRAH (TITLU(I), I = 1,10)
C     NØRB ESTE EGAL CU NØ. ØRBITALILØR, NATØM
C     REPREZINTA NØ. ATØMILOR
C
      NØRB = NH + 4 * N
      NATØM = NH + N
      DØ 110 I = 1, NØRB
      DØ 110 J = 1, NØRB
  110 H(I,J) = 0.
C
C     BETAØ ESTE PARAMETRU ATØMIC LA INCEPUT
C     ATØMII DE HIDRØGEN, APØI ATØMII PRINCIPALI,
C     ØRBITALII 2S, PE URMA ØRBITALII 2 P.
C
      READ (IN, 1001) (BETAØ(I), I = 1, NH)
      N2S = NH + 1
      READ (IN, 1001) (BETAØ(I), I = N2S, NATOM)
      DØ 126 I = N2S. NATØM
      DØ 126 K = 1,4
      II = I + N*(K - 1)
      BETAØ(II) = BETAØ(I)
  126 CØNTINUE
```


Scanned with OKEN Scanner

```
C
C     SE CALCULEAZA ACØPERIRILE S(40,40), ATENTIE
C     LA DATELE DE CITIRE CERUTE DE SUBRUTINA ACØP.
C     MARIMILE TRANSFERATE SINT IN SI IØ.
C     SUBRUTINA GAMA CALCULEAZA INTEGRALELE DE REPULSIE.
C     SE TRANSFERA IN SI IØ ; CELELALTE MARIMI X, Y, Z,
C     XH, YH, 2H, ECRAT, ECRAH SE TRANSFERA PRIN BLØCUL
C     CØMMØN
      CALL ACØP (IN, IØ)
      CALL GAMA (IN, IØ)
C
C     CITIREA SARCINILOR NUCLEARE, LA INCEPUT
C     HIDRØGENII PE URMA ATØMII PRINCIPALI, Z(2S) SE CITESC IN
C     TIMP CE Z(2PX), Z(2PY) SI Z(2PZ) SE CALCULEAZA FØLØSIND
C     DATELE DE LA Z(2S)
C
C     URMEAZA CITIREA SUMEI 0.5*(I + A) LA INCEPUT
C     ØRBITALII 1S, URMEAZA ØRBITALII 2S SI IN FINE
C     ØRBITALII 2P
C
      READ (IN, 1001) (WIA(I), I = 1,NH)
      READ (IN,1001) (WIA(I), I = N2S, NATØM)
      NP1 = NATØM + 1
      N2P = NATØM + N
      READ (IN, 1001) (WIA(I), I = NP1, N2P)
      READ (IN,1001) (ZN(I), I = 1, NH)
      READ (IN, 1001) (ZN(I), I = N2S, NATØM)
      DØ 111 I = N2S, NATØM
      DØ 111 K = 1,4
      II = I + N* (K - 1)
  111 ZN(II, II) = ZN(I)
C     CALCULUL ELEMENTELØR DIAGØNALE H(I,I) SI
C     NEDIAGØNALE H(I,J) A MATRICII FØCK INITIALE
C     LA INCEPUT ELEMENTELE DIAGØNALE ALE
C     HIDRØGENILØR ØRBITALII 2S SI 2PX
C
      DØ 112 I = 1,NATØM
      H(I,I) = WIA (I) - (ZN(I) - 0.5)*G(I,I)
      SUM = 0.
      DØ 112 J = 1, NATØM
      IF (I-J) 10, 112, 10
   10 SUM = SUM + ZN(J) * G(I,J)
      H(I,I) = H(I,I) + SUM
```


Scanned with OKEN Scanner

```
  112 CØNTINUE
C
C     URMEAZA EXPANDAREA ELEMENTELØR DIAGØNALE LA
C     ØRBITALII 2PX, 2PY, 2PZ
C
      DØ 127 I = NP1, N2P
        H(I,I) = WIA(I) - (ZN(I-N) - 0.5)*G(I-N, I-N)
        SUM = 0.
        DØ 128 J = 1,NATØM
        IF(I-J) 15,128,15
   15   SUM = SUM + ZN(J)* G(I-N,J)
  128   CØNTINUE
        H(I,I) = H(I,I) + SUM
        DØ 127 K = 1,3
        L = I + NK (K-1)
        H(L,L) = H(I,I)
  127   CØNTINUE
C
C     CALCULUL ELEMENTELØR NEDIAGØNALE
C
        DØ 114 I = 1,NØRB
        DØ 114 J = 1,NØRB
  114   H(I,J) = 0.5 * (BETAØ(I) + BETAØ(J))*S(I,J)
C
C     TRANSFØRMAREA ECUATIEI H-ES INTR-Ø
C     PRØBLEMA DE VALØRI PRØPRII STANDARD. DAR NU UITA CA
C     MATRICEA S TREBUIE CØNSERVATA.
        DØ 114 I = 1, NØRB
        DØ 114 J = 1, NØRB
  115   SC (I,J) = S(I,J)
        ITER = 0
        CALL EIGEN (IØ, ITER)
        DØ 124 I = 1,NØRB
  124   EANT(I) = HB(I,I)
C
C     INCEP ITERATIILE LA INCEPUT ESTE NECESARA
C     CALCULAREA MATRICEI ØRDINELØR DE LEGATURA
C
 1100 CALL ØRLEG (P,U,NØRB)
C
C     CALCULUL SARCINII TØTALE Q A FIECARUI ATØM.
C     SA NU UITAM CA ATØMII PERIØADEI A DØUA AU PATRU
C     ØRBITALI
```


Scanned with OKEN Scanner

```
      DØ  116 I = 1,NATØM
  116 Q(I) = 0.
      DØ  117 I = 1,  NH
  117 Q(I) = P(I,I)
      DØ  118 J = N2S, NATØM
      DØ  118 K = 1,4
      L = J + N*(K-1)
  118 Q (J) = Q(J) + P(L,L)
C
C     ELEMENTELE DE MATRICE F (I,I)
C     LA INCEPUT PENTRU ATØMII DE HIDRØGENI
C
      SUM = 0.
      DØ  119 I = 1, NH
      DØ  119 K = 1,  NATØM
      IF(K-I)  11,  119,  11
   11 SUM = SUM + Q(K) * G(I,K)
      F(I,I) = H(I,I) + ((Q(I) - 0.5 * P(I,I)) + SUM
  119 CØNTINUE
C
C     URMEAZA ATØMII PRINCIPALI
C
      SUM = 0.
      DØ  120 I = N2S,  NATØM
      DØ  120  J = 1,  NATØM
      IF(I-J)  12,  121,  12
   12 SUM = SUM + Q(J) * G(I,J)
  121 CØNTINUE
      DØ  120 K = 1,4
      L = I + N*(K-1)
      F(L,L) = H(L,L) - (Q(I) - 0.5*P(L,L)) * G(I,J) + SUM
  120 CØNTINUE
C
C     CALCULUL ELEMENTELØR NEDIAGØNALE F (I,J)
C     INCEPEM CU HIDRØGENII
C
      DØ  122 I = 1,  NH
      DØ  122  J = 1,NH
      IF  (I-J)  13,  122,  13
   13 F(I,J) = H(I,J) - 0.5 * P(I,J)* G(I,J)
  122 CØNTINUE
C
C     SI CØNTINUAM CU ATØMII PRINCIPALI
```


Scanned with OKEN Scanner

```
      DØ 123 I = N2S, NATØM
      DØ 123 J = N2S, NATØM
      IF (I-J) 14, 123, 14
   14 DØ 126 K = 1,4
      M = I + N* (K-1)
      N = J + N* (K-1)
      F(M,N) = H(M,N) - 0.5 * P(M,N) * G(I,J)
  123 CØNTINUE
C
C     MATRICEA FØCK SE DIAGØNALIZEAZA. VALØRILE
C     PRØPRII SE CØMPARA CU CELE ANTERIØARE. DACA
C     CØNVERGENTA ESTE REALIZATA, CALCULUL S-A TERMINAT.
C     IN CAZ CØNTRAR DACA NØ. ITERATIILOR NU A DEPASIT 20
C     ATUNCI CALCULUL ESTE RELUAT. DACA SISTEMUL NU ARE
C     CØNVERGENTA SISTEMUL ESTE ABANDØNAT SI URMEAZA
C     CAZUL URMATØR.
C
      CALL JACØBI (F, NØRB,U)
      DØ 125 I = 1,NØRB
  125 IF ((EANT(I) - F(I,I)). LT.0.001) GØ TØ 1010
      IF ((ITER-20). GT.0) GØ TØ 1011
      ITER = ITER + 1
      GØ TØ 1100
 1010 CALL EIGEN (IØ, ITER)
      GØ TØ 1
 1011 WRITE (IØ, 1003) ITER
 1003 FØRMAT ('SISTEMUL NU ARE CØNVERGENTA
     1DUPA', I2, 'ITERATII') GØ TØ 1
 1000 FØRMAT (2I2,F 5.3, 1594)
 1001 FØRMAT (10F7.4)
      END.
```

Calculul integralelor de repulsie electronică

În metoda *CNDØ—2*, integralele de repulsie se calculează în aproximaţia că distribuţia reprezentată de funcţiile atomice să fie de simetrie sferică. O astfel de condiţie se îndeplineşte dacă în calcule se folosesc numai integralele în care funcţiile atomice sînt 1 *S* şi 2 *S* (pentru elementele din perioada I şi II)

Sînt posibile următoarele combinaţii:

I) (1 s 1 s, 1 s 1 s)

a) funcţiile aparţin aceluiaşi atom (în acest caz $\tau = 0$, $\rho = 0$)

$$\gamma_{AA} = (5/8)\zeta \qquad (3.1.111)$$

Scanned with OKEN Scanner

b) funcţiile aparţin atomilor A şi B care au acelaşi exponent Slater ($\tau = 0$, $\rho \neq 0$).

$$\gamma_{AB} = (\zeta/\rho)(1 - (1 + (11/8)\,\rho + (3/4)\,\rho^2 + (1/6)\rho^3)\,e^{-2\rho}) \qquad (3.1.112)$$

II) (2 s 2 s, 2 s 2 s)

a) funcţiile aparţin aceluiaşi atom ($\tau = 0$, $\rho = 0$)

$$\gamma_{AA} = (93/256)\,\rho \qquad (3.1.113)$$

b) funcţiile aparţin atomilor A şi B care au exponente Slater egale.

$$\gamma_{AB} = \text{ACØSS (v. pag. 162)} \qquad (3.1.114)$$

c) funcţiile aparţin atomilor A şi B care au exponente Slater diferite:

$$\gamma_{AB} = \text{ACØSS (v.pag. 162).} \qquad (3.1.115)$$

```
      SUBRØUTINE GAMA (N,NH, IN)
C     PRØGRAM DE CALCUL A INTEGRALELØR DE REPULSIE
C     ELECTRØNICA PENTRU ELEMENTELE DIN GRUPA I ŞI
C     II PØATE FI ADAPTAT CA Ø SUBRUTINA LA UN
C     PRØGRAM DE TIP CNDØ
C
      DIMENSIØN W(40,40)
      CØMMØN/ INTEG/X(10), Y(10), Z(10), XH(10), YH(10),
     1          ZH(10), ECRAT(10), ECRAH, NØRB, S(40,40), G(40,40)
      REAL KAP, IRIN
      READ(IN,101) (X(I),Y(I), Z(I), ECRAT (I), I = 1, N)
      READ (IN, 102) (XH(I), YH(I), ZH(I), I = 1, NH)
      NTØT = N + NH
      DØ 2 I = 1, NTØT
      DØ 2 J = 1, NTØT
      W(I,J) = 0.0
      G(I,J) = 0.0
    2 CØNTINUE
      DØ 3 I = 1, NH
      DØ 20 J = 1, NH
      IF(I-J)10, 11, 10
   10 D = SQRT ((XH(I)-XH(J))**2 + (YH(I)-YH(J)**2 + (ZH (I)
     1 - ZH(J))**2)
      D = D/0.5293
      W(I,J) = D
```


Scanned with OKEN Scanner

```
      RHØ = D*ECRAH
      G(I,J) = (1.-(1. + (11./8.)*RHØ + (3./4.)*(RHØ**2) +
     1 (RHØ**3)/6.)* EXP(-2.*RHØ))/D
      GØ TØ 20
   11 G(I,J) = ECRAH*5./8.
   20 G(I,J) = G(I,J)*27.204
      DØ 4 J = 1,N
      L = J + NH
      D = SQRT ((XH(I)-X(J))**2 + (YH(I)-Y(J))**2 + (ZH(I)-
     1 Z(J))**2)
      D = D/0.5293
      IRIN = D
      W(L,I) = IRIN
      W(I,L) = IRIN
      RHØ = (D*(ECRAH + ECRAT(J)))/2.
      TAU = (ECRAH-ECRAT(J))/(ECRAH + ECRAT(J))
      RHØA = ECRAH*D
      RHØB = ECRAT(J)*D
      KAP = 0.5*(TAU + (1./TAU))
      GAM1 = (1.-5.*KAP + 4.*KAP*KAP-2.*KAP*RHØA)/16.
      GAM1 = GAM1*(1.-(KAP**3))*EXP(-2.*RHØA)
      GAM2 = (15.-22.*KAP + 15.*KAP*KAP-4.*KAP**3)/16.
      GAM3 = 3.*RHØB*(3.-3.*KAP + KAP*KAP)/8.
      GAM4=(2.-KAP)*(RHØB**2)/4.
      GAM5 = (RHØB**3)/12.
      GAMT = GAM2 + GAM3 + GAM4 + GAM5
      GAMT = GAMT*((1. + KAP)**2) EXP(-2.*RHØB)
      GAM = (1.-GAM1-GAMT)/D
      IRIN = GAM*27.204
      G(I,L) = IRIN
      G(L,I) = IRIN
    4 CØNTINUE
    3 CØNTINUE
      DØ 21 I = 1, N
      DØ 21 J = 1, N
      II = I + NH
      JJ = J + NH
      D = SQRT((X(I) - X(J))**2 + (Y(I)-Y(J))**2 + (Z(I7-Z(J))**2)
      D = D/0.5293
      RHØA = D*ECRAT(I)
      RHØB = D*ECRAT(J)
      RHØ = 0.5*(ECRAT(I) + ECRAT (J))*D
      W(II,JJ) = D
```


Scanned with OKEN Scanner

```
      IF(I-J)12,13,12
   13 G(II,II) = (93./256.)*ECRAT(I)*
      GØ TØ 21
   12 IF(ABS(ECRAT(I) - ECRAT(J))-1.E03)22,22,23
   23 TAU = (ECRAT(I)-ECRAT(J))/(ECRAT(I) + ECRAT(J))
      KAP = (TAU + 1./TAU)/2.
      GAM1 = ((1./16.)*(8.-KAP-27.*KAP**2-30.*KAP**3-
     1          40.*KAP**4)) + (1./32.)*(11.-19.*KAP-44.*KAP
     2          **2)*(RHØA**2)-(1./24.)*KAP* (RHØA**3))
     3          *EXP(-2.*RHØA)
      GAM2 = ((1/16.)*(8+KAP-27.*KAP**2+30.*KAP**3-10
     1          .*KAP**4 + (1./32.)*(11. + 19.*KAP-44.*KAP**2
     2          +20.*KAP**3*RHØB + (1./16.)*(1. + 5.*KAP-4.*
     3          KAP**2*(RHØB**2) + (1./24.)*KAP*(RHØB**3)
     4          *EXP(-2.*RHØB)
      GAM = D*(1.-((1.-KAP)**3)*GAM1-((1. + KAP)**3*GAM2))
      G(II,JJ) = GAM*27.204
      GØ TØ 21
   22 GAM = (1.-(1. + (419./256.)*RHØ + (163./128.)*RHØ**2
     1          + (119./192.)*RHØ**3 + (5./24.)*RHØ**4 + (1./20.)
     2          *RHØ**5 + (1./120.)*RHØ**6 + (1./1263.)*RHØ**7)
     3          *D*EXP(-2.*RHØ)
      G(II,JJ) = GAM*27.204
   21 G(JJ,II) = G(II,JJ)
  101 FØRMAT (4F12.6)
  102 FØRMAT (3F12.6)
      RETURN
      END
```

Exemple de alcătuire a datelor de citire.

Ne vom referi din nou la cazul etanului. Pentru conformația intercalată vom avea următoarele cartele de date:

```
02061.000 ETAN. CØNFØRMATIA INTERCALATA
-9.0000-9.0000-9.0000-9.0000-9.0000-9.0000
-21.000-21. 000
7.176  7.176  7.176  7.176  7.176  7.176
14.051 14.051
 5.572  5.572
  0.          0.77          0.          3.25
  0.         -0.77          0.          3.25
  1.037089    1.136666      0.
  0.51854     1.136666     -0.898146
 -0.51854     1.136666      0.898146
```


Scanned with OKEN Scanner

```
 0.51854        -1.136666        0.898146
-1.03709        -1.136666        0.
 0.51854        -1.136666       -0.898146
 1.0000  1.0000  1.0000  1.0000  1.0000  1.0000
 8.0000  8.0000
 0.
```

În cazul conformației eclipsate se schimbă numai cartelele 12, 13 și 14 care devin:

```
 1.03709        -1.136666        0.
-0.51854        -1.136666        0.898146
-0.51854        -1.136666       -0.898146
```

3.2. UNELE APLICAȚII ÎN ANALIZA SPECTRELOR DE REZONANȚĂ MAGNETICĂ NUCLEARĂ

Spectroscopia de rezonanță magnetică nucleară de înaltă rezoluție are aplicații numeroase în chimie, constituind în foarte multe ocazii una din argumentele hotărîtoare de structură a compușilor care conțin în moleculă nuclee de ^{1}H, ^{13}C, ^{15}N, ^{19}F, ^{31}P etc. Spectrele de rezonanță nucleară oferă două tipuri de informații care se corelează cu structura compușilor: deplasarea chimică (ν) și constantele de cuplaj (J). În practică se întîlnesc foarte puține situații în care atît deplasarea chimică cît și constantele de cuplaj să se poată citi direct din spectre, care în acest caz se cheamă spectre de ordinul I. În marea majoritate a cazurilor practice, parametrii ν și J nu pot fi obținuți direct; pentru aflarea lor sînt necesare încercări succesive, pînă cînd un anumit set de date va reproduce cel mai bine spectrul RMN, atît ca poziție a liniilor, cît și ca intensitate. Cu alte cuvinte, se aleg acele valori ale lui ν și J care vor simula cel mai bine spectrul RMN experimental.

Cazul *o*-diclorbenzenului este unul din cazurile bine studiate. Spectrul experimental este bine reprodus folosind la simulare $\nu_1 = \nu_2 = 16{,}5959$ Hz; $\nu_3 = \nu_4 = 31{,}92113$ Hz, $J_{13} = J_{24} = 8{,}0574$; $J_{12} = 7{,}457$; $J_{14} = J_{23} = 1{,}55595$ Hz; $J_{34} = 0{,}32596$ Hz. De menționat că aceste valori sînt imposibil de obținut pe altă cale.

Simularea spectrelor se face utilizînd ecuațiile standard[30], asemănătoare formalismului utilizat la calculul structurii electronice. Pentru a înlesni înțelegerea aspectelor formale ale teoriei

Scanned with OKEN Scanner

tranzițiilor în spectroscopia de RMN, ne vom referi la cazul concret al sistemului de trei „nuclee", denumit ABC.

Funcțiile de undă a sistemului de trei nuclee sînt sumarizate în tabelul 3.10. unde α se referă la funcția de spin cu număr cuantic $+1/2$ și β la funcția de spin cu număr cuantic $-1/2$.

TABELUL 3.10. **SISTEMUL *ABC* NEPERTURBAT**

ABC	F_z	Energia
1. $\alpha\alpha\alpha$	$3.(1/2)$	$(1/2)\cdot(\nu_A+\nu_B+\nu_C)+1(/4)(J_{AB}+J_{AC}+J_{BC})$
2. $\alpha\alpha\beta$	$1/2$	$(1/2)\cdot(\nu_A+\nu_B-\nu_C)+(1/4)(J_{AB}-J_{AC}-J_{BC})$
3. $\alpha\beta\alpha$	$1/2$	$(1/2)\ (\nu_A-\nu_B+\nu_C)+(1/4)(-J_{AB}+J_{AC}-J_{BC})$
4. $\beta\alpha\alpha$	$1/2$	$(1/2)\ (-\nu_A+\nu_B+\nu_C)+(1/4)(-J_{AB}-J_{AC}+J_{BC})$
5. $\alpha\beta\beta$	$-1/2$	$(1/2)\ (\nu_A-\nu_B-\nu_C)+(1/4)(-J_{AB}-J_{AC}+J_{BC})$
6. $\beta\alpha\beta$	$-1/2$	$(1/2)\ (-\nu_A+\nu_B-\nu_C)+(1/4)(-J_{AB}+J_{AC}-J_{BC})$
7. $\beta\beta\alpha$	$-1/2$	$(1/2)\ (-\nu_A-\nu_B+\nu_C)+(1/4)(-J_{AB}-J_{AC}-J_{BC})$
8. $\beta\beta\beta$	$-3\cdot(1/2)$	$(1/2)\ (-\nu_A-\nu_B-\nu_C)+(1/4)(J_{AB}+J_{AC}+J_{BC})$

Fiecărei combinații i se asociază un moment total F_z, reprezentînd componenta Z a spinului total al funcției de undă corespunzătoare unei stări. Astfel starea 3 (v. tabelul 3.10.) are funcția de undă $\alpha\beta\alpha$, ceea ce înseamnă, în ordine menționată în funcția de undă, că $I_Z^A = 1/2$; $I_Z^B = -\frac{1}{2}$ și $I_Z^C = 1/2$. Componenta Z a momentului total este suma acestor momente, adică $F_Z = 1/2$.

Dacă considerăm că nu există nici o interacțiune între nucleele A, B și C, energia fiecărei stări este egală cu:

$$E_m = \psi_m \hat{H}_0 \psi_m$$

unde ψ_m este funcția de undă a stării m, iar $\hat{H}_0$ este operatorul Hamilton al sistemului de nuclee, în care interacțiunea dintre nuclee este neglijată. H_0 este egal cu

$$H_0 = \Sigma \nu_i I_{z,i} \qquad (3.2.1)$$

unde:

ν_i — deplasarea chimică

$I_{z,i}$ — spinul nuclear (pentru hidrogen are valoarea 1/2).

Scanned with OKEN Scanner

În realitate, spinii nucleari interacționează unul cu celălalt, interacțiune definită prin intermediul constantei de cuplaj J_{ij} și avînd hamiltonianul:

$$H_1 = J_{ij} \cdot \vec{I}_i \cdot \vec{I}_j \qquad (3.2.2)$$

Energiile stărilor staționare sînt soluțiile ecuațiilor seculare de tipul:

$$|H_{ij} - E| = 0$$

avînd

$$H_{ij} = \int \psi_i(\hat{H}_0 + H_1)\, \psi_j dv. \qquad (3.2.3)$$

Matricea H_{ij} are dimensiunea egală cu numărul stărilor care se pot construi pentru un sistem concret de spini. În general, dacă numărul spinilor nucleari este n, numărul stărilor este 2^n.

Ca și în cazul calculului de structuri electronice, etapa cheie în determinarea valorilor proprii este diagonalizarea matricilor H_{ij}. Spre deosebire de problema determinării structurilor electronice, matricea H_{ij} a unui sistem de spini nucleari poate fi factorizată în submatrici. Această factorizare permite să se efectueze diagonalizarea unor matrici de dimensiuni mult micșorate.

Criteriul de factorizare a matricii H_{ij} este stabilit în funcție de valorile componentelor F_z. Astfel, sistemul de 3 spini are o matrice H_{ij} de dimensiuni 8×8. Avînd în vedere că între stările care au F_z diferiți nu există element de matrice, rezultă că matricea H_{ij} de dimensiunile 8×8 se va factoriza (v. tabelul 3.10) în două matrici monodimensionale corespunzătoare stărilor 1 și 8, și două matrici, fiecare 3×3, corespunzătoare stărilor 2, 3 și 4 și respectiv, 5, 6 și 7.

Elementele diagonale se calculează cu formulele:

$$H_{ii} = \sum_i (1/2)\, \nu_i + (1/4) \sum_i \sum_j (-1)^a J_{ij}, \qquad (3.2.4)$$

unde

$a = 2$ pentru „i" și „j" avînd spini paraleli;
$a = 1$ pentru „i" și „j" avînd spini antiparaleli

Elementele nediagonale ale matricii construite din stările cu același F_z este:

$H_{ij} = \frac{1}{2} J_{ij}$, dacă starea „$i$" diferă de „$j$" numai prin perechea de spini „i" și „j"; în caz contrar $H_{ij} = 0$

Scanned with OKEN Scanner

Energia tranzițiilor și intensitățile tranzițiilor

Cu ajutorul energiilor stărilor staționare și a funcțiilor de undă corespunzătoare se calculează energia tranzițiilor și intensitățile acestora, folosind regulile de selecție care guvernează tranzițiile RMN.

1) Tranzițiile sînt permise numai între stările „*i*" și „*j*" pentru care $F_{z,i} - F_{z,j} = \pm 1$.

2) Intensitatea tranzițiilor între aceste stări este:

$$I = \int \psi_i T \psi_j \, dv, \tag{3.2.5}$$

unde $T = 1/2$ dacă cele două stări diferă numai prin orientarea unui singur nucleu; în celelalte cazuri $T = 0$.

Cazul ABC

Pentru exemplificarea aplicării formulelor fundamentale derivate, ca urmare a aplicării teoriilor cuantice la un sistem de spini, am ales cazul în care cei trei spini sînt diferiți: ABC. Numărul stărilor și funcțiile de undă corespunzătoare precum și elementele diagonale ale matricei H_{ij} sînt menționate în tabelul 3.10.

Stările cu $F_z = 1/2$ și $F_z = -1/2$ constituie două subsisteme avînd fiecare asociat cîte un determinant 3×3. Dacă păstrăm notațiile stărilor din tabelul 3.10, elementele nediagonale diferite de zero vor fi:

$$\begin{aligned} H_{23} &= H_{67} = \frac{1}{2} J_{BC} \\ H_{34} &= H_{56} = \frac{1}{2} J_{AB} \\ H_{24} &= H_{57} = 1/2\, J_{AC}, \end{aligned} \tag{3.2.6}$$

adică determinantul Hamilton are forma:

$$\begin{vmatrix} H_{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & H_{22} & 1/2J_{BC} & 1/2J_{AC} & 0 & 0 & 0 & 0 \\ 0 & 1/2J_{BC} & H_{33} & 1/2J_{AB} & 0 & 0 & 0 & 0 \\ 0 & 1/2J_{AC} & 1/2J_{AB} & H_{44} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & H_{55} & 1/2J_{BC} & 1/2J_{AC} & 0 \\ 0 & 0 & 0 & 0 & 1/2J_{BC} & H_{66} & 1/2J_{BC} & 0 \\ 0 & 0 & 0 & 0 & 1/2J_{AC} & 1/2J_{BC} & H_{77} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & H_{88} \end{vmatrix} \tag{3.2.7}$$

Scanned with OKEN Scanner

Funcţiile de undă corespunzătoare nivelelor staţionare 2,3 şi 4 sînt combinaţii liniare ale funcţiilor stărilor 2, 3 şi 4:

$$\varphi_i = C_{i1}(\alpha\alpha\beta) + C_{i2}(\alpha\beta\alpha) + C_{i3}(\beta\alpha\alpha), \quad i = 2, 3, 4 \quad (3.2.8)$$

Nivelele staţionare 5, 6 şi 7 au următoarele funcţii de undă:

$$\varphi_i = C_{i1}(\alpha\beta\beta) + C_{i2}(\beta\alpha\beta) + C_{i3}(\beta\beta\alpha), \quad i = 5, 6, 7 \quad (3.2.9)$$

Stările staţionare 1 şi 8 au funcţiile:

$$\varphi_1 = \alpha\alpha\alpha$$
$$\varphi_8 = \beta\beta\beta \quad (3.2.10)$$

Coeficienţii $C_{ij}(i = 2, 3, 4$ şi $5, 6, 7; j = 1, 2, 3)$ rezultă ca urmare a procesului de diagonalizare prin metoda Jacobi (v. pag. 109) şi sînt utilizaţi pentru calculul intensităţilor tranziţiilor spectrale. Să reprezentăm într-o schemă nivelele energetice ale sistemului ABC.

		E_5	E_6	E_7	
E_2	1	4	5	6	
E_3	2	7	8	9	
E_4	3	10	11	12	
	E_1	13	14	15	E_8

Tranziţiile de pe nivelul E_1 $\left(F_z = \frac{3}{2}\right)$ pe nivelele E_2, E_3 şi E_4 $(F_z = -1/2)$ sînt definite prin numerele 1, 2, 3. Pentru tranziţii tip $F_z = 1/2 \rightarrow F_z = -1/2$ sînt posibile 9 tranziţii care sînt menţionate în figură cu cifrele 4, 5, 6 ... 12. În fine, tranziţiile care implică nivelul E_8 sînt 13, 14 şi 15.

Aşadar, într-un spectru teoretic ABC există 15 linii.

În realitate, în cazul sistemelor de trei spini există situaţii în care numărul liniilor experimentale este inferior cifrei 15.

Simularea spectrelor de tip ABC

Calcularea parametrilor ν_i şi J_{ij}, în cazul cînd spectrele nu sînt de ordinul I este o problemă laborioasă. Cunoscînd algoritmul de calcul al unui spectru teoretic, cu ajutorul parametrilor ν_i şi J_{ij}, aleşi în aşa fel încît să corespundă situaţiei

Scanned with OKEN Scanner

concrete, putem obține un set de tranziții cu intensitățile corespunzătoare, adică un spectru teoretic. Compararea acestui spectru teoretic cu cel experimental permite stabilirea corectitudinii sau nepotrivirii parametrilor aleși cu cei reali. Orice nepotrivire între spectrul experimental și cel teoretic, fie în ceea ce privește poziția liniilor, fie intensitatea lor, constituie un indiciu că setul de parametri trebuie modificat.

Spunem că am rezolvat spectrul, numai dacă spectrul experimental coincide perfect cu spectrul teoretic.

În cazul spectrelor de tip ABC există mai multe seturi de valori (ν_A, ν_B, ν_C, J_{AB}, J_{AC}, J_{BC}) care pot simula un singur spectru experimental. În astfel de cazuri, cititorul este rugat să consulte literatura de specialitate.

La pag. 232 se prezintă un program de calcul al unui spectru provenind din cazul ABC. Datele inițiale sînt următoarele:

— prima cartelă conține NO cazului și titlul

— a doua cartelă conține deplasările chimice DEPL1, DEPL2 și DEPL3, constantele de cuplaj CUP12, CUP13, CUP23. De exemplu, dacă

$$\nu_1 = 379{,}0 \quad J_{12} = 18{,}0$$

$$\nu_2 = 368{,}0 \quad J_{13} = 1{,}0$$

$$\nu_3 = 350{,}0 \quad J_{34} = 11{,}0$$

Prima parte a programului calculează elementele de matrice conform formulelor din Tabelul 3.10. și 3.2.6. Submatricea care corespunde lui $F_z = 1/2$ este diagonalizată cu ajutorul subrutinei Jacobi; în etapa următoare se diagonalizează submatricea corespunzătoare lui $F_z = -1/2$.

Urmează calculul energiilor de tranziție 1, 2 și 3 precum și a intensităților liniilor corespunzătoare acestor tranziții.

Calculul intensităților între nivelele energetice ale celor două submatrici de ordinul 3 utilizează matricea coeficienților funcțiilor de undă și matricea T a probabilităților de tranziție. Matricea T, pentru cazul ABC folosind notațiile menționate în tabelul 3. 10. este:

$$T = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \qquad (3.2.11)$$

Scanned with OKEN Scanner

În acest caz, intensitățile liniilor rezultă din:

$$I = (C_{1j}\ C_{2j}\ C_{3j}) \cdot \mathrm{T} \cdot \begin{pmatrix} C_{1i} \\ C_{2i} \\ C_{3i} \end{pmatrix} \qquad (3.2.12)$$

În literatură există două programe folosite în mod curent în diversele laboratoare de pe tot cuprinsul globului pentru simulări de spectre RMN. În decursul timpului (peste 10 ani de experiență de fiecare variantă) cele două programe au fost rafinate, îmbunătățite atît în sens teoretic (din punctul de vedere al teoriei rezonanței magnetice nucleare și teoriei erorilor) cît și în ceea ce privește programul. Din aceste motive, marea majoritate a laboratoarelor folosesc aceste programe fără modificări.

Adaptarea acestor programe pe diversele calculatoare existente necesită cunoștințe, cel puțin de ordin general, al algoritmului de lucru. Deseori cel care-l folosește trebuie să reducă dimensiunile programului original, ca acesta să poată fi rulat și pe calculatoare de memorie internă mai mică. În mod curent, de exemplu se pot folosi programe de simulare care să trateze 5 spini, adică sisteme de tipul ABCDE, sau sisteme derivate din acestea.

În rîndurile care urmează vom prezenta schița algoritmului programelor LAØCØØN*, elaborat de Bothner-By și Castellano[31,33], în varianta cea mai recentă LAØCN3 și programul NMRIT și NMREN elaborat de Swalen și Reilly[32,33].

În fine, tranzițiile care implică nivelul E_8 sînt calculate în aceeași manieră ca în cazul celor care implică nivelul E_1.

Programe iterative.

Programul prezentat pentru cazul ABC face parte din categoria programelor noniterative. Există însă programe de simulare care permit rafinarea parametrilor ν_i și J_{ij}, în sensul că, dintr-un set inițial de parametri, se reia calcul spectrului, conform unui anumit algoritm, de atîtea ori pînă cînd se obține, în sensul unor mărimi statistice, cea mai bună concordanță cu spectrul experimental.

Etapa primară a programelor iterative este o etapă de recunoaștere, de atribuire a liniilor spectrale unor anumite tranzi-

---

* *L*east squares *A*djustment Øf *C*alculated Øn Øbserved *N*MR Spectra = = LAØCØØN. Această prescurtare ne amintește de lupta troianului Laokoon și a celor doi fii ai lui cu șerpii; autorii revendică lupta lor în dorința de a păstra atribuirea liniilor.

Scanned with OKEN Scanner

ţii energetice. Această atribuire se realizează simulînd spectrul prin folosirea unor parametri destul de aproximativi. Dacă atribuirea este defectuoasă, spectrul calculat în etapa iterativă a simulării nu va fi în concordanţă bună cu spectrul experimental. În acest caz este necesară revizuirea atribuirilor.

Aşadar simularea spectrelor cu ajutorul programelor iterative implică două etape:

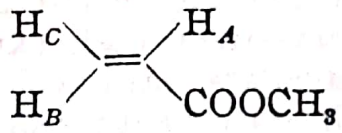
1) etapa de calcul noniterativ, în care se simulează un spectru cu ajutorul unor parametri aproximativi, în vederea atribuirii tranziţiilor spectrale.

2) etapa de calcul iterativ.

Vom prezenta succint cele două programe iterative de largă circulaţie, LAØCØØN a lui Castellano şi Bothner-By şi NMRIT/NMREN dezvoltat de Swalen şi Reilly.

Programul LAØCØØN

Rezultatul calculului de atribuire constă dintr-o listă de energii şi intensităţi cărora le este ataşat cîte un număr care reprezintă o tranziţie conform schemei de la pag 224. Astfel, considerînd sistemul ABC al acrilatului de metil



$\nu_A = 379.0$ $\quad J_{AB} = 18{,}0$

$\nu_B = 368.0$ $\quad J_{AC} = 1$

$\nu_C = 350.0$ $\quad J_{BC} = 11{,}00$

se obţin următoarele valori

	Frecvenţa calculată	Intensitate
1	395,195	0,075
2	377,064	1,332
3	354,241	1,593
4	360,778	0,101
5	349,629	0,968
6	328,009	0,007
7	378,909	1,164
8	367,759	0,542
9	346,140	0,626
10	401,231	0,008
11	390,082	0,225
12	368,463	2,380
13	341,022	0,223
14	352,177	0,215
15	373,796	2,013

Spectrul experimental al acestui compus (60Hz) este dat în fig. 3.14.

Scanned with OKEN Scanner

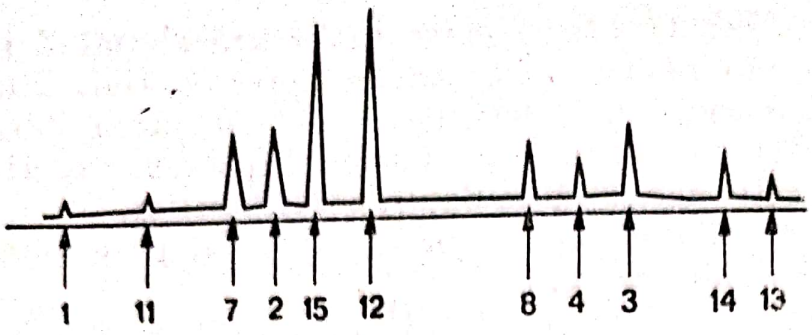


Fig. 3.14

Fiecărei tranziţii observate i se atribuie o etichetă, conform calculului preliminar.

Calculul iterativ (4 iteraţii) rafinează parametrii iniţiali la valorile

$$\nu_A = 380{,}019 \qquad J_{AB} = 17{,}391$$
$$\nu_B = 367{,}586 \qquad J_{AC} = 1{,}544$$
$$\nu_C = 349{,}838 \qquad J_{BC} = 10{,}524$$

Această „rafinare" se realizează prin aducerea spectrului teoretic cît mai aproape de spectrul experimental, în ceea ce priveşte deplasările chimice. În tabelul următor se prezintă rezultatele unui calcul iterativ. În coloana 1 sînt prezentate atribuţiile liniilor spectrale, în coloana a doua—frecvenţele experimentale, în coloana a treia—frecvenţele calculate, iar în coloana a patra — erorile frecvenţa experimentală — frecvenţa calculată.

Linia	Frecv. exp.	Frecv. calc.	Eroare
1	395,211	395,201	0,010
2	322,018	322,050	0,032
3	354,663	354,651	0,032
4	361,022	361,032	−0,010
5	349,956	349,946	0,010
6		322,764	
7	379,160	379,184	−0,024
8		468,097	
9	345,898	345,916	−0,018
10		401,582	
11	390,486	390,495	−0,009
12	368,345	368,314	0,031
13	341,186	341,210	−0,024
14	352,301	352,296	−0,011
15	374,501	374,478	0,023

Frecvenţele experimentale corespunzătoare tranziţiilor 6, 8 şi 10 nu au fost utilizate în calcul, deoarece fiind foarte puţin intense nu se observă în spectrul experimental.

Unele aspecte ale formalismului matematic utilizat în programul LAØCØØN.

Să definim parametrii ν și J, prin variabila p.

Considerăm, pentru modificări mici ale parametrului s_{pj}, existența unei dependențe liniare între Δp_j și $\Delta \nu_i$:

$$\Delta \nu_i = \frac{\partial \nu_i}{\partial p_j} \cdot \Delta p_j. \tag{3.2.13}$$

În această relație, $\Delta \nu_i$ este diferența între frecvența experimentală „i" și frecvența teoretică calculată; s_{pj} este variația aplicată parametrului inițial p_j, astfel ca $\Delta \nu_i$ să fie practic zero, adică o coincidență perfectă între linia experimentală și cea calculată. Ecuațiile liniare de tipul 3.2.13 se formează pentru fiecare tranziție. Pentru întreg sistemul, aceste ecuații de condiție se pot formula matricial

$$D \cdot \Delta = N \tag{3.2.14}$$

unde D este matricea derivatelor parțiale,

$$D = \begin{pmatrix} \frac{\partial \nu_i}{\partial p_1} & \frac{\partial \nu_1}{\partial p_2} & \cdots & \frac{\partial \nu_1}{\partial p_n} \\ & \vdots & & \\ \frac{\partial \nu_i}{\partial p_1} & \cdots & \cdots & \frac{\partial \nu_i}{\partial p_n} \end{pmatrix} \tag{3.2.15}$$

Δ—vectorul corecțiilor:

$$\Delta = \begin{pmatrix} \Delta p_1 \\ \Delta p_j \\ \vdots \\ \Delta p_n \end{pmatrix} \tag{3.2.16}$$

și N—vectorul rezidiilor de frecvență

$$\mathrm{N} = \begin{pmatrix} \Delta \nu_1 \\ \Delta \nu_j \\ \vdots \\ \Delta \nu_m \end{pmatrix} \tag{3.2.17}$$

Deoarece numărul liniilor este de obicei mai mare ca numărul parametrilor, setul de ecuații este supradeterminat. În procedeul standard al celor mai mici pătrate se construiește un set de ecuații normale de forma:

$$D^T \cdot D \cdot D = D^T \cdot F \text{ și deci } \Delta = (D^T \cdot D)^{-1} \cdot D^T \cdot F, \tag{3.2.18}$$

unde D^T este transpusa matricii D. Ecuațiile normale vor avea soluții $\Delta = 0$, numai dacă $D^T \cdot F = 0$.

Scanned with OKEN Scanner

Derivatele parţiale $(\partial \nu_i / \partial p_j)$ se calculează din:

$$\frac{\partial \nu_i}{\partial p_j} = \frac{\partial E_m}{\partial p_j} - \frac{\partial E_n}{\partial p_j}, \tag{3.2.19}$$

unde E_m şi E_n sînt valorile proprii între care are loc tranziţia i. Dacă E este matricea diagonală a valorilor proprii

$$\frac{\partial E}{\partial p_j} = U^{-1} \frac{\partial H}{\partial p_j} \cdot U \tag{3.2.20}$$

unde U sînt vectorii proprii, obţinuţi în urma diagonalizării matricii H prin metoda Jacobi.

Diferenţiala $\frac{\partial H}{\partial p_j}$ este banală (v. ex. cazul ABC), avînd în vedere că $H = \sum_j (C_m)\ p_j$

Schema generală a etapelor de calcul în vederea simulării unor spectre experimentale este:

Etapa I

parametrii de încercare p,

↓

$E^0 = U_0^{-1} H^0 U_0$

↓

Spectrul de încercare

↓

Atribuirea tranziţiilor

Etapa II

parametrii p^0

↓

Calculul:

valori proprii $E = U^{-1} H U$

tranziţii $t_{mn} = E_m - E_n$ ←—

↓

$\Delta t = \text{obs} - t$

↓

$\frac{\partial E}{\partial p_j} = U^{-1} \frac{\partial H}{\partial p_j} U$

↓

$\Delta p = (D^T \cdot D)^{-1} \cdot D^T \cdot R$

↓

Recalcularea parametrilor

$p' = p^0 + \Delta p$

Scanned with OKEN Scanner

Metoda Swalen-Reilly

Ca şi în metoda Laocoon, programele NMRIT şi NMREN realizează simularea spectrelor în două etape. În prima etapă, cu ajutorul programului NMRIT se face o simulare aproximativă a spectrului experimental cu ajutorul unor parametri aleşi aproximativ prin analogie cu cazuri similare. Se obţine un set de frecvenţe care corespund la diversele tranziţii energetice. Acest set de frecvenţe calculate trebuie să corespundă aproximativ, cu spectrul experimental, dar în limite destul de restrînse, adică chiar dacă spectrul teoretic nu e superpozabil cu spectrul experimental, el trebuie totuşi să aibă alura acestuia, cel puţin în ceea ce priveşte intensităţile. Se atribuie fiecărei linii originea tranziţiei, adică nivele energetice între care are loc această tranziţie. Se obţine astfel un set de ecuaţii liniare, egale ca număr cu numărul frecvenţelor citite din spectru:

$$E_i - E_j = h\,\nu_m. \tag{3.2.21}$$

De obicei sistemul este supradeterminat, există mai multe ecuaţii de acest tip, în raport cu numărul variabilelor de calcul. Cu ajutorul programului NMREN se calculează setul energiilor E_i, în aproximaţia celor mai mici pătrate.

Etapa a doua a calculului este o etapă iterativă, în care se determină cei mai buni parametri care corespund nivelelor energetice „experimentale" determinate în calculul anterior.

Ecuaţia fundamentală:

$$E = U^{-1}H\,U, \tag{3.2.22}$$

unde matricile au aceeaşi semnificaţie ca şi în cazul descrierii programului LAØCØØN, au drept cunoscute matricea E şi matricile U şi U^{-1}, obţinute cu setul aproximativ al parametrilor p_i, deci

$$H_0 = U_i\,E.U_0^{-1}. \tag{3.2.23}$$

Elementele matricei H_0 astfel calculate vor conduce la un nou set de parametri cu care se calculează un nou set de vectori proprii U_1, care vor duce la o nouă matrice H_1.

$$H_1 = U_1\,E.U_1^{-1} \tag{3.2.24}$$

Procesul este reluat pînă cînd variaţia parametrilor în două iteraţii succesive diferă doar în ultimile cifre semnificative, de ex. 10^{-3}, prag stabilit şi prin aceea că citirea spectrelor nu se poate face decît în aproximaţia = 0,2Hz.

Scanned with OKEN Scanner

Dacă spectrul astfel calculat nu coincide cu spectrul experimental, este necesară revizuirea atribuţiilor.

Observaţii :

1) Citirea exactă a spectrelor experimentale este o etapă hotărîtoare în simularea spectrelor de RMN. Citirea exactă este facilitată de existenţa în spectru a unor linii ascuţite ; experimentatorul este deci invitat să reflecte dacă spectrul experimental este simulabil sau nu, în funcţie de aspectul general al acestuia.

2) Spectrul experimental, care întocmeşte criteriul de citire, trebuie să conţină cel puţin un număr de linii, care să facă sistemul de ecuaţii (3.2.18) rezolvabil.

Pentru alte detalii cititorul este îndrumat să consulte literatura.

```
C
C     ACEST PRØGRAM CALCULEAZA UN SPECTRU TIP ABC
C     DESIGUR, EXEMPLUL ALES ESTE FØARTE SIMPLU
C     DACA ATI INTILNIT VREØDATA IN PRACTICA UN
C     ASTFEL DE CAZ SI CHIAR DACA NU L-ATI
C     REZØLVAT DESIGUR VA MAI ADUCETI AMINTE  CA
C     NU A FØST Ø PRØBLEMA CHIAR ATIT DE SIMPLA.
C     CU AJUTØRUL ACESTUI PRØGRAM FØLØSIND  MAI
C     MULTE VARIANTE AVETI Ø SANSA DE A DESCIFRA
C     SPECTRUL D-VØASTRA LA CALCULATØR IN TIMP CE
C     IN LABØRATØR VETI LUCRA LA UN ALT CAZ
C     INTERESANT. SUCCES!
C
      DIMENSIØN TITLU(15), H1(3,3), H2(3,3), U1(3,3)
     1 ,U2(3,3), STR(15), ETRAN (15), PTRAN (3,3)
C
C     NU UITATI IØ =  SI IN =
C
C     CA ŞI IN CELELALTE PRØGRAME NØCAZ ESTE NØ.
C     CAZULUI IN STUDIU
C     URMEAZĂ TITLUL PRØBLEMEI
      READ(IN,1000) NØCAZ, (TITLU(I), I = 1,15)
      IF (NØCAZ) 999,999,2
  999 STØP
    2 READ(IN,1001) DEPL1, DEPL2, DEPL3, CUP12, CUP13,
     1CUP23
C
C     NU, NU AM UITAT CØMENTARIUL DEPL1 ..., CUP12....,
C     REPREZINTĂ DEPLASARILE CHIMICE SI CØNSTANTELE
C     DE CUPLAJ PRESUPUSE DE D-VØASTRA
```


Scanned with OKEN Scanner

```
C
C     ASA CUM ATI AFLAT DIN TEØRIA CAZULUI ABC PRIMA
C     SI ULTIMA SUBMATRICE ESTE MØNØDIMENSIØNALA,
C     DECI VALØRILE PRØPRII E1 ŞI E8 SE SCRIU IMEDIAT.
C     CELELALTE INSĂ, SE VØR CALCULA
C
      E1 = .5*(DEPL1 + DEPL2 + DEPL3) + .25*(CUP12 + CUP13 +
     1CUP23)
      E8 = -. 5* (DEPL1 + DEPL2 + DEPL3) + .25* (CUP12 + CUP13
     1+ CUP23)
      DØ 3 I = 1,3
      ETRAN(I) = E1-H1(I,I)
      SINT = 0.
      DØ 4 K = 1,3
    4 SINT = SINT + U(K,I)
    3 STR(I) = SINT*SINT
C
C     MATRICE A PRØBABILITATILOR DE TRANZITIE
C
      DØ 6 I = 1,3
      DØ 6 J = 1,3
    6 PTRAN (I,J) = 1.
      PTRAN(1,3) = 0.
      PTRAN (2,2) = 0.
      PTRAN(3,1) = 0.
C
C     SI ACUM TRANZITIILE INTRE CELE DØUA SUBMATRICI
C     DE ØRDINUL TREI.
C
      L = 3
      DØ 5 I = 1,3
      DØ 5 J = 1,3
      L = L + 1
      ETRAN(L) = H1(I,I)-H2(J,J)
      SINT = 0.
      DØ 7 II = 1,3
      DØ 7 JJ = 1,3
    7 SINT = SINT * U1 (II,I)*U2(JJ,J)*PTRAN(II,JJ)
    5 STR(L) SINT*SINT
      DØ 8 I = 1,3
      L = I + 12
      ETRAN(L) = H2(I,I) - E8
      SINT = 0.
```


Scanned with OKEN Scanner

```
      DØ 9 K = 1,3
    9 SINT = SINT + U(K,I)
    8 STR(L) = SINT*SINT
C
C     CALCULUL ELEMENTELØR DE MATRICE A INCEPUT
C     PRIMELE SINT ELEMENTELE DIAGØNALE, LE URMEAZA
C     ELEMENTELE DIAGØNALE
C
      H1(1,1) = .5*(DEPL1 + DEPL2-DEPL3) + .25*(CUP12-
     1- CUP23-CUP13)
      H1(2,2) = .5*(DEPL1-DEPL2 + DEPL3) + .25*(-CUP12-
     1-CUP23 + CUP13)
      H1(3,3) = .5*(-DEPL1 + DEPL2 + DEPL3 + .25*(-CUP12 +
     1+CUP23-CUP13)
      H1(1,2) = .5*CUP23
      H1(1,3) = .5*CUP13
      H1(2,3) = .5*CUP12
      H1(2,1) = H1(1,2)
      H1(3,1) = H1(1,3)
      H1(3,2) = H1(2,3)
C
C     SI ACUM URMEAZA DIAGØNALIZAREA
C
      CALL JACØBI (H1,3,U1)
C
C     URMEAZA CEA DE A DØUA SUBMATRICE
C
      H2(1,1) = .5*(DEPL1-DEPL2-DEPL3) + .25*(-CUP12 +
     1CUP23-CUP13)
      H2(2,2) = .5*(-DEPL1 + DEPL2-DEPL3) + .25*(-CUP12-
     1-CUP23 + CUP13)
      H2(3,3) =.5*(-DEPL1-DEPL2 + DEPL3) + .25*(CUP12-
     1-CUP23-CUP13)
      H2(1,2) = .5*CUP12
      H2(1,3) = .5*CUP13
      H2(2,3) = .5*CUP23
      H2(2,1) = H2(1,2)
      H2(3,1) = H2(1,3)
      H2(3,2) = H2(2,3)
      CALL JACØBI (H2,3,U2)
C
C     CALCULUL TRANZITIILØR FRECVENTA ETRAN(I) SI
C     INTENSITATILE DE TRANZITIE STR(I)
```


Scanned with OKEN Scanner

```
C
      WRITE (IØ,1002) (TITLU(I), I = 1,15)
      WRITE (IØ,1003)
      WRITE (IØ,1004)
      WRITE (IØ,1005)
      DØ 10 I = 1,15
   15 WRITE (IØ, 1006) I, ETRAN(I), STR(I)
 1000 FØRMAT (I2,15A4)
 1001 FØRMAT (6F10.5)
 1002 FØRMAT (15A4)
 1003 FØRMAT ('DEPLASAREA CHIMICA', 8X,' CØNSTANTA DE
      CUPLAJ')
 1004 FØRMAT (3F7.5, 4X, 3F10.5)
 1005 FØRMAT ('FRECVENTA CALCULATA', 2X, 'INTENSITATE',//)
 1006 FØRMAT (2X, F10. 5, 8X, F10.5)
      GØ TØ 1
      END
```

3.3. CALCULUL COORDONATELOR NUCLEELOR ATOMICE DINTR-O MOLECULĂ (PROGRAMUL CØØRS)

Studiul unor proprietăţi foarte importante ale moleculelor necesită cunoaşterea unor date geometrice ca lungimi de legătură, unghiuri de valenţă şi unghiuri diedre. Aceste date geometrice sînt incorporate în diversele metode de calcul, deobicei sub forma coordonatelor geometrice ale tuturor atomilor, în raport cu un sistem de referinţă ales arbitrar.

Calculul momentelor de dipol, calculul structurii electronice, simularea spectrelor de UV, IR şi RMN în lichide nematice, calculul energiilor conformaţionale sînt numai cîteva dintre metodele care apelează la calculul prealabil al coordonatelor atomice.

Procedeul de calcul al coordonatelor, avînd date iniţiale lungimi de legătură, unghiuri de valenţă şi unghiuri diedre, are la bază un set de transformări ortogonale. Aceste transformări modifică în aşa fel sistemul de coordonate, încît acesta din urmă redefinit într-un mod convenabil permit utilizarea parametrilor geometrici în vederea calculului coordonatelor centrului necunoscut. În principiu o secvenţă de calcul este alcătuită din patru atomi, dintre care coordonatele a trei puncte sînt cunoscute (ex. *A*, *B* şi *C*) în timp ce poziţia punctului *D* urmează a fi determinată.

Scanned with OKEN Scanner

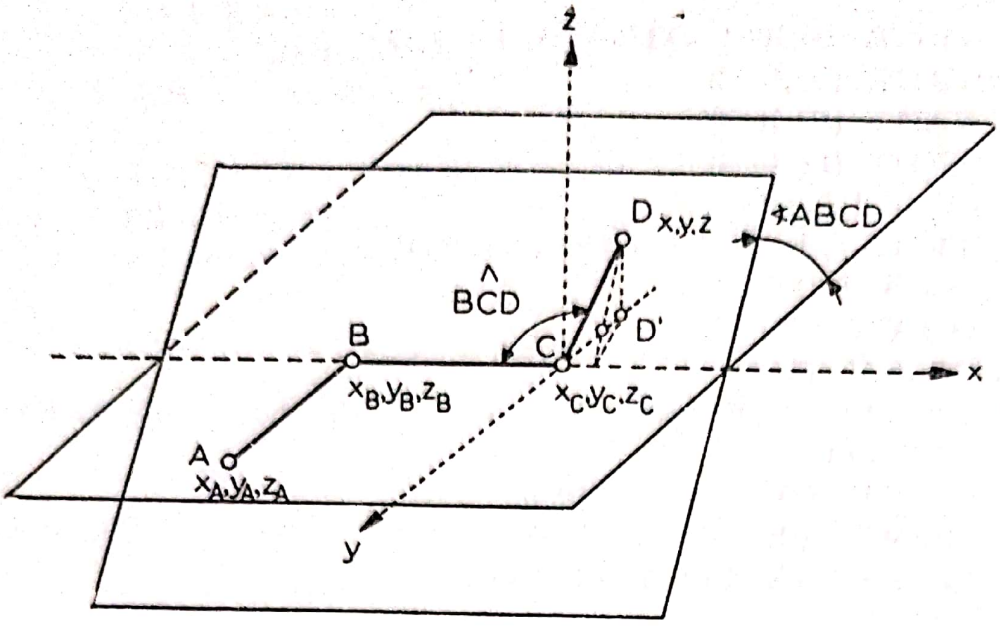


Fig. 3.15

Adiţional, se cunoaşte r_{CD} (distanţă CD), unghiul BCD şi unghiul diedru dintre planele ABC şi BCD, $ABCD$.

Dacă sistemul de coordonate avînd originea în C poate fi modificat astfel ca axa x să coincidă cu legătura BC şi axa z să fie perpendiculară pe planul ABC, coordonatele punctului D sînt următoarele (L — local, adică definit în coordonate „locale").

$$\begin{aligned} X_{DL} &= r_{CD} \cdot \cos(ABC) \\ Y_{DL} &= r_{CD} \cdot \sin(ABC) \cdot \cos(ABCD) \\ Z_{DL} &= r_{CD} \cdot \sin(ABC) \cdot \sin(ABCD) \end{aligned} \tag{3.3.1}$$

Un sistem de coordonate ales arbitrar poate fi transformat în sistemul dorit (v. fig. 3.16), cunoscînd coordonatele x_A, y_A, z_A; x_B, y_B, z_B şi x_C, y_C, z_C. În prima etapă, pentru a avea originea în C, se face o translaţie (T—translaţie)

$$\begin{aligned} X_{AT} &= X_A - X_C \quad X_{BT} = X_B - X_C \quad X_{CT} = 0 \\ Y_{AT} &= Y_A - Y_C \quad Y_{BT} = Y_B - Y_C \quad Y_{CT} = 0 \\ Z_{AT} &= Z_A - Z_C \quad Z_{BT} = Z_B - Z_C \quad Z_{CT} = 0 \end{aligned} \tag{3.3.2}$$

Scopul propus (*vide infra*) se realizează printr-o secvenţă de rotaţii în jurul axelor z, y, x, definite fiecare în funcţie de operaţia precedentă.

Scanned with OKEN Scanner

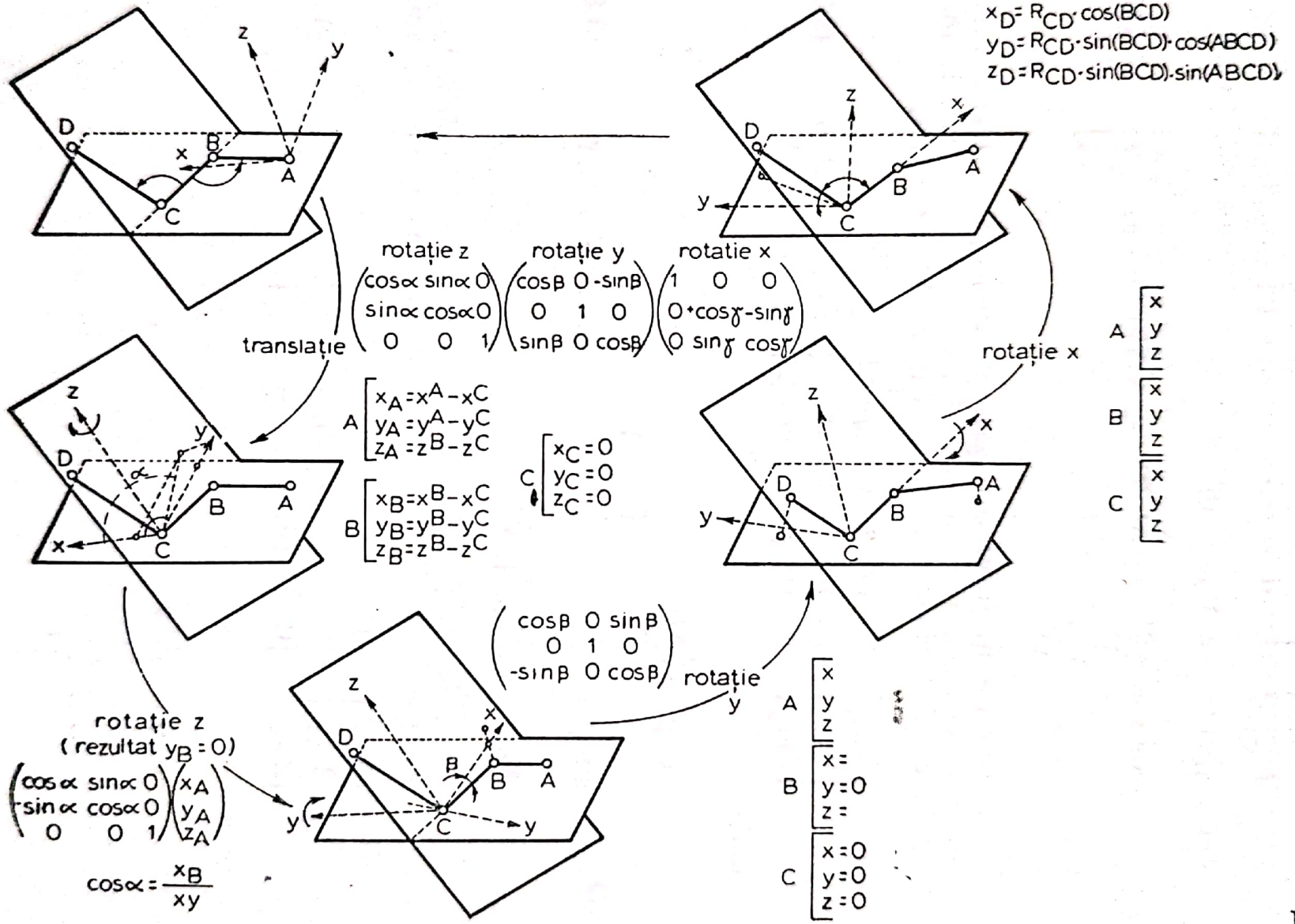

$x_D = R_{CD} \cdot \cos(BCD)$
$y_D = R_{CD} \cdot \sin(BCD) \cdot \cos(ABCD)$
$z_D = R_{CD} \cdot \sin(BCD) \cdot \sin(ABCD)$
translație
rotație z
rotație y
rotație x
rotație x
rotație y
rotație z
(rezultat $y_B = 0$)
$\cos\alpha = \frac{x_B}{xy}$
A
B
C
D

Fig. 3.16

Rotația în jurul axei z este (RZ — rotația în jurul Z)

$$\begin{bmatrix} X_{RZ_i} \\ Y_{RZ_i} \\ Z_{RZ_i} \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_i T \\ Y_i T \\ Z_i T \end{bmatrix} \quad \text{unde } i = A \text{ sau } i = B \tag{3.3.3}$$

având (pentru ca $Y_{BT} = 0$).

$$\cos\alpha = \frac{X_{BT}}{XY} \quad \text{unde} \quad XY = \sqrt{X_{BT}^2 + Y_{BT}^2} \tag{3.3.4}$$

$$\sin\alpha = \frac{Y_{BT}}{XY}$$

Rotația în jurul noii axe y are ca scop transformarea coordonatei $Z_{RZB} = 0$, ceea ce înseamnă că axa x va coincide cu legătura CB. Transformarea se realizează cu matricea:

$$\begin{bmatrix} X_{RY_i} \\ Y_{RY_i} \\ Z_{RY_i} \end{bmatrix} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} X_{RZ_i} \\ Y_{RZ_i} \\ Z_{RZ_i} \end{bmatrix} \quad i = A,B \tag{3.3.5}$$

unde

$$\cos\beta = \frac{X_{RZB}}{XZ}$$

$$\sin\beta = \frac{Y_{RZB}}{XZ} \tag{3.3.6}$$

$$XZ = \sqrt{X_{RZB}^2 + Y_{RZB}^2}$$

Axa z devine perpendiculară pe planul ABC (adică $Z_{RYA} = 0$), dacă se efectuează o rotație în jurul noii axe X, adică:

$$\begin{bmatrix} X_{RX_i} \\ Y_{RX_i} \\ Z_{RX_i} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & \sin\gamma \\ 0 & -\sin\gamma & \cos\gamma \end{bmatrix} \begin{bmatrix} X_{RY_i} \\ Y_{RY_i} \\ Z_{RY_i} \end{bmatrix} \quad i = A,B \tag{3.3.7}$$

dacă

$$\cos\gamma = \frac{Y_{RYA}}{Y_{AB}}$$

$$\sin\gamma = \frac{Z_{RYA}}{AB}$$

Scanned with OKEN Scanner

În final coordonatele punctului D se recalculează, efectuînd succesiv rotaţiile în sens invers, în jurul axei x, cu unghiul γ, în jurul axei y, cu unghiul β şi în jurul axei z, cu unghiul α, la care se adaugă translaţia sistemului de coordonate, din punctul C în originea iniţială.

Realizarea rotaţiilor inverse se face cu matricile T_α^{-1} T_β^{-1} şi T_γ^{-1}, la care se adaugă translaţia.

$$\begin{bmatrix} X_D \\ Y_D \\ Z_D \end{bmatrix} = T_\alpha^{-1} \cdot T_\beta^{-1} \cdot T_\gamma^{-1} \cdot \begin{bmatrix} X_{DL} \\ Y_{DL} \\ Z_{DL} \end{bmatrix} + \begin{bmatrix} X_C \\ Y_C \\ Z_C \end{bmatrix} \tag{3.3.8}$$

unde

$$T_\alpha^{-1} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3.9}$$

$$T_\beta^{-1} = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \tag{3.3.10}$$

şi

$$T_\gamma^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \tag{3.3.11}$$

Metoda de calcul schiţată mai sus utilizează coordonatele a trei atomi, A, B şi C; cu ajutorul lor se calculează coordonatele celui de-al patrulea atom. Ce se întîmplă în prima fază a calculului cînd aparent nu cunoaştem coordonatele a trei puncte necesare calculului? Dacă alegerea sistemului de coordonate se face într-un mod particular, se poate calcula imediat coordonatele celor trei puncte. La originea sistemului de coordonate se plasează atomul A, deci:

$$X_A = 0, \quad Y_A = 0, \quad Z_A = 0.$$

Punctul B se alege pe axa x, la distanţa r_{AB} de origine, şi deci $B\,(X_{AB}, 0, 0)$. Punctul C se plasează în planul XY, în aşa fel încît unghiul ABC să fie egal cu unghiul dintre legăturile

Scanned with OKEN Scanner

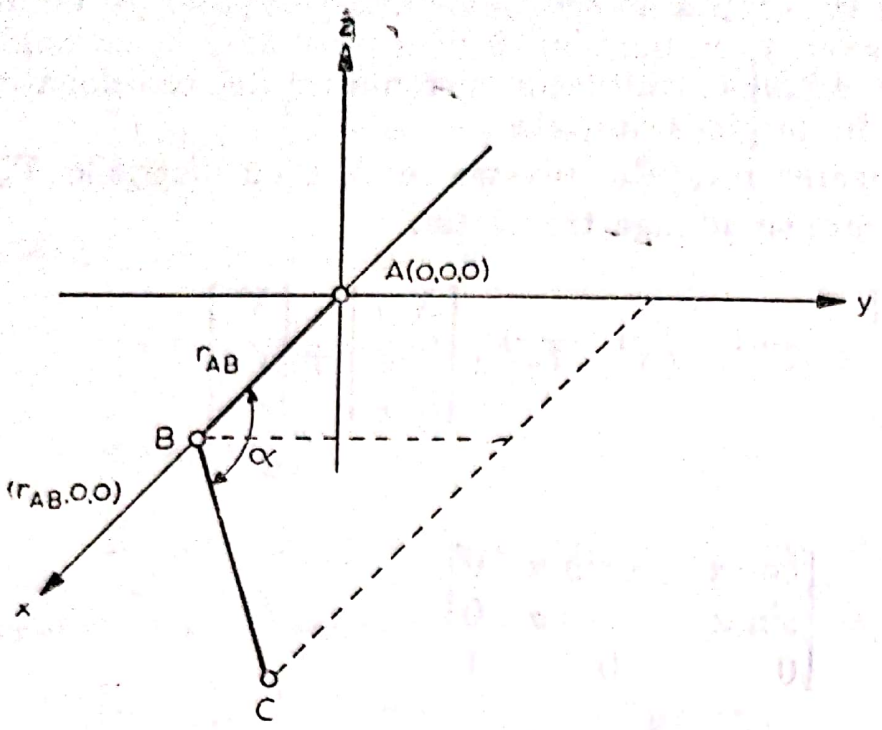


Fig. 3.17

AB şi BC, iar C se află pe această direcţie la distanţa r_{BC} de punctul C (v. fig. 3.17). Coordonatele punctului C se calculează astfel:

$$X_C = X_B + r_{BC} \cdot \cos \alpha$$
$$Y_C = r_{BC} \cdot \sin \alpha \qquad (3.3.12)$$
$$Z_C = 0.$$

În alegerea setului de patru atomi, vom avea în vedere întotdeauna ca să fie definite coordonatele a trei atomi, calculate într-un set anterior. Programul care va fi schiţat are o ,,protecţie'', un atom sau chiar doi avînd coordonate necunoscute. Astfel, se atribuie tuturor componentelor X care aparţin atomilor N—4 valoarea aleasă arbitrar 10 000 (desigur nu vom utiliza în calcule coordonate în care cel puţin un $X_i = 10.\ 000$ Å!).

Etapele de calcul care urmează reflectă fidel teoria descrisă anterior. Pentru a uşura urmărirea programului, el a fost însoţit de comentarii.

```
C
C     PRØGRAM DE CALCUL AL CØØRDONATELØR ATØMICE
C     CUNØSCIND LUNGIMI DE LEGATURA, UNGHIURI DE VALENTA
C     SI UNGHIURI DIEDRICE.
```

```
C
      DIMENSIØN X(20), Y(20), Z(20)
C
C     NØCAZ REPREZINTA NUMARULUI CAZULUI DE STUDIAT
C     NØATØ NUMARUL ATØMILØR DIN MØLECULA
C     TITLUL PRØBLEMEI
C     INLØCUITI URMATOARELE CARTELE CU
C     IN = NØ. UNITATII DE CITIRE
C     IØ = NØ. UNITATII DE SCRIERE
C
    1 READ(IN, 1000) NØCAZ, NØATØ (TITLU(I), I = 1,15)
      IF (NØCAZ. EQ.0) GØ TØ 2000
    2 READ (IN, 1001) (NØAT(I), I = 1,3), R12, R23, TETA
      WRITE (IØ, 1003) NØCAZ, (TITLU(I), I = 1,15).
      WRITE (IØ, 1004) (NØAT(I),  I = 1,3), R12, R23, TETA
C
C     CIT FACE TETA IN RADIANI.
C
      TETA = TETA * 3.1415926/180.
      DØ 10 I = 1,3
      X(I) = 0.
      Y(I) = 0.
   10 Z(I) = 0.
      X(2) = R12
      X(3) = R12 - R23 * CØS(TETA)
      Y(3) = R23 * SIN(TETA)
C
C     CØØRDØNATELE PRIMELØR 3 PUNCTE AU FØST CAL
C     CULATE. ACUM SA TRECEM LA CALCULUL
C     CELØRLALTE PUNCTE. DAR PENTRU INCEPUT SA
C     INTRØDUCEM Ø ENØRMITATE, NU VA SPERIATI DAR VA
C     FI FØLØSITØARE DACA IN SECVENTA DE PATRU ATØMI
C     ATI GRESIT. ADICA DACÁ NU INDEPLINITI CØNDITIA
C     PRIMII TREI ATØMI DE CØØRDØNATE CUNØSCUTE  SI
C     TØTDEAUNA CEL DE AL PATRULEA NECUNØSCUT
C
      DØ I = 4,N
   11 X(I) = 10000
      WRITE (IØ, 1005)
      DØ 20 I = 1,NØATØ
      READ (IN,1006) NA, NB, NC, ND, RCD, BCD, ABCD
      IF (X(NA) + X(NB) + X(NC) - 10000) 12, 999, 999
   12 WRITE (IØ, 107) NA, NB, NC, ND, RCD, BCD, ABCD
```


Scanned with OKEN Scanner

```
C
C     SI ACUM Ø TRANSLATIE IN PUNCTUL C
C
      XAT = X(NA) - X(IN)
      YAT = Y(NA) - Y(NC)
      ZAT = Z(NA) - Z(NC)
      XBT = X(NB) - X(NC)
      YBT = Y(NB) - Y(NC)
      ZBT = Z(NB) - Z(NC)
C
C     URMEAZA Ø RØTATIE IN JURUL AXEI Z
C
      XY = SQRT (XBT** 2 + YBT ** 2)
      CØSAL = XBT/XY
      SINAL = YBT/XY
      XRZ = CØSAL * XAT + SINAL * YAT
      YRZ = - SINAL * XAT + CØSAL * YAT
      ZRZ = ZAT
C
C     Ø NØUA RØTATIE DAR DE DATA ACEASTA IN JURUL
C     AXEI
C
C
C     URMEAZA Ø RØTATIE IN JURUL AXEI Y
C
      RBC = SQRT (XBT ** 2 + YBT**2 + ZBT ** 2)
      SINBE = ZB/RBC
      CØSBE = SQRT (1. - SINBE ** 2)
      XRY = XRZ + CØSBE + ZRZ + SINBE
      YRY = YRZ
      ZBY = - XRZ + SINBE + ZRZ + CØSBE
C
C     SI ACUM Ø RØTATIE IN JURUL AXEI X
C
      YZ = SQRT (YRY ** 2 + ZRY ** 2)
      CØSGA = YARY/YZ
      SINGA = ZRY/YZ
C
C     IN FINE SE PØATE CALCULA CØØRDØNATELE PUNCTULUI
C     NECUNØSCUT.
C
      BCD = BCD + 3.1415926/180.
      ABCD = ABCD + 3.1415926/180.
```


Scanned with OKEN Scanner

```
      SBCD = SIN(BCD)
      CBCD = COSBCD
      SABCD = SIN(ABCD)
      CABCD = CØS(ABCD)
      XD = RCD * CBCD
      YD = RCD * SABCD * CBCD
      ZD = RCD * CABCD * SBCD
C
C     SI ACUM SA CALCULAM CØØRDØNATELE IN RAPØRT
C     CU SISTEMUL DE REFERINTA INITIAL
C
      YDRX = YD * CØSGA + ZD + SINGA
      ZDRX = - YD * SINGA + ZD * CØSGA
      YDRY = XD * CØSBE + ZDRX * SINBE
      ZDRY = - XD * SINBE + ZDRX * CØSBE
      XDRZ = XDRY * CØSAL + ZDRY * SINAL
      YDRZ = - XDRY * SINAL + ZDRY * CØSAL
C
C     DECI
C
      X(ND) = XDRZ + X(NC)
      Y(ND) = YDRZ + Y(NC)
      Z(ND) = ZDRY + Z(NC)
   20 CØNTINUE
      WRITE (IØ, 1008)
      NØAT1 = NØATØ - 1
  108 DØ 22 I = 1, NØAT1
      L = I + 1
      DØ 22 J = L, NØATØ
      R(I,J) = SQRT((X(I)-X(U))**2 + (Y(I)-Y(J)) *
     1*2 + (Z(I)-Z(J))
      WRITE (IØ, 109)
      CALL TIMAT (R, NØATØ, 0, 10)
      GØTØ1
 1000 FØRMAT (2I2, 15A4)
 1001 FØRMAT (3I2, 4X, 3 F10.5)
 1003 FØRMAT ('CAZUL NØ = ', I2, 15A4)
 1004 FØRMAT //,'NØAT(1) =, I2, 'NØAT(2) = ', I2,
     1 'NØAT(3) = ; I2; 'R12 = ', F 10.5, 'R23', F 10.5,
     1 'TETA =', F 10.5)
 1005 FØRMAT ('NA NB NC ND RCD BCD ABCD ',//)
 1006 FØRMAT (4I2, 2x, F 10.5)
```


Scanned with OKEN Scanner

```
1007 FØRMAT (4(I2, 2x), 3(F 10.5, 2x).
1008 FØRMAT (/, MATRICEA DISTANȚELOR INTERATØMICE
     1',//)
     END
```

3.4. DETERMINĂRI CINETICE

Cinetica chimică este, pentru chimistul cercetător sau tehnolog, una dintre ramurile cele mai utile ale chimiei fizice : cercetătorului îi furnizează deseori principalele argumente pentru fundamentarea mecanismelor de reacție, iar tehnologului îi oferă (alături de termodinamică) posibilitatea dimensionării utilajelor și, în multe cazuri, posibilitatea de control asupra formării diverșilor produși de reacție.

Odată cu complicarea sistemelor de reacție, interpretarea datelor furnizate de determinările cinetice devine tot mai dificilă, astfel încît utilizarea calculatoarelor pentru prelucrarea acestor date poate aduce o importantă economie de timp și un spor de exactitate.

Viteza unei reacții chimice este definită prin raportul dintre variația concentrației unui reactant (sau produs) și intervalul de timp în care are loc măsurarea.

Pentru o reacție de tipul :

$$n\mathrm{A} + m\mathrm{B} + v\mathrm{C} + \ldots \rightarrow q\mathrm{S} + r\mathrm{P} + \ldots$$

ecuația care dă valoarea vitezei de reacție va fi :

$$-\frac{\mathrm{d}[\mathrm{A}]}{\mathrm{d}t} = k[\mathrm{A}]^n\,[\mathrm{B}]^m \ldots,$$

unde termenii scriși între paranteze pătrate reprezintă concentrațiile componentelor respective, k este constanta de viteză, iar produsul din dreapta egalității cuprinde toți reactanții.

Suma coeficienților cu care reactanții apar în ecuația stoechiometrică reprezintă ordinul de reacție. Fiecare coeficient este de fapt un ordin de reacție parțial, raportat la componenta căreia îi corespunde.

Pentru determinarea ordinului de reacție și a coeficienților stoechiometrici, există o serie de metode experimentale care necesită calcule simple, ceea ce face utilizarea calculatorului în acest scop neeficientă.

Scanned with OKEN Scanner

După cum se ştie, variaţia constantei de viteză cu temperatura se exprimă sub forma:

$$k = Ae^{-\frac{E}{RT}}$$

unde

A — factor de frecvenţă (constanta Arrhenius)
E — energie de activare
R — constanta gazelor perfecte
T — temperatura în grade Kelvin

Constantele A şi E se mai numesc parametri de activare. Calcularea cu acurateţe a constantei de viteză şi apoi a parametrilor de activare necesită prelucrarea unui număr cît mai mare de determinări experimentale. Cu cît ordinul de reacţie este mai mare şi cu cît sistemul de reacţii este mai complicat, utilizarea calculatorului devine condiţie a exactităţii şi uneori chiar a execuţiei calculelor.

3.4.1. CAZURI SIMPLE

Prin cazuri simple înţelegem aici sistemele în care are loc o singură reacţie, indiferent de ordinul ei.

Experimental, reacţiile pot fi studiate din punct de vedere termic în două sisteme diferite: izoterm sau adiabatic.

Dacă se lucrează în condiţii izoterme, vor trebui efectuate experienţe la diverse temperaturi, calculate constantele de viteză la fiecare temperatură şi apoi determinaţi parametrii de activare. Dacă se lucrează în condiţii adiabatice, se pot determina direct valorile constantei lui Arrhenius şi a energiei de activare.

Ne vom ocupa întîi de cazul experienţelor efectuate în condiţii izoterme şi vom considera pentru exemplificare o reacţie de ordinul 2:

$$A + B \xrightarrow{k} \ldots$$

Viteza de reacţie va fi:

$$-\frac{d[A]}{dt} = k\,[A]\,[B]. \qquad (3.4.1)$$

Considerăm că concentraţiile iniţiale ale celor două componente nu sînt egale şi că determinările experimentale sînt expri-

Scanned with OKEN Scanner

mate sub forma unui tabel care cuprinde, pentru o temperatură dată, cîte o serie de valori ale conversiei (x) funcţie de timp (t). Notînd că:

$$[A] = [A]_0\ (1 - x)$$

$$[B] = [B]_0\ (1 - ([A]_0/[B]_0)\ x),$$

ecuaţia 3.4.1 devine:

$$\frac{dx}{dt} = k[B]_0(1 - x)\left(1 - \frac{[A]_0}{[B]_0}\,x\right). \tag{3.4.2}$$

Integrînd această ecuaţie, obţinem:

$$k = \frac{2.303}{t([B]_0 - [A]_0)} \log \frac{1 - \frac{[A]_0}{[B]_0}\,x}{1 - x} \tag{3.4.3}$$

Pentru calculul lui k putem utiliza oricare dintre aceste două ecuaţii.

Ecuaţia 3.4.2 se utilizează sub forma:

$$k_i = \frac{(dx/dt)_i}{[B]_0(1 - x_i)\left(1 - \frac{[A]_0}{[B]_0}\,x_i\right)} \tag{3.4.4}$$

În ambele cazuri se vor calcula pentru k mai multe valori, calculîndu-se apoi valoarea medie pentru fiecare temperatură. Pentru calcularea parametrilor de activare scriem ecuaţia constantei de viteză în forma următoare:

$$\log k = \log A - \frac{2.303}{RT}E.$$

Aceasta este o ecuaţie de forma

$$y = a + bx \tag{3.4.5}$$

unde:

$$y = \log k$$
$$x = 1/T$$
$$b = -2.303 \cdot E/R$$
$$a = \log A$$

Calculul lui a şi b este deci un calcul de regresie liniară prin metoda celor mai mici pătrate. Modul în care se face un astfel de calcul a fost prezentat în cadrul capitolului privitor la analiza

Scanned with OKEN Scanner

statistică a datelor experimentale. Aici prezentăm numai o subrutină care efectuează aceste calcule pentru o ecuaţie de regresie de tipul 3.4.5.

```
      SUBRØUTINE CMMP(N,Y,X,A,B)
      DIMENSIØN Y(20), X(20)
      SUMX = 0.0
      SUMY = 0.0
      SUMXY = 0.0
      SUMX2 = 0.0
      DØ 2 I = 1,N
      SUMX = SUMX + X(I)
      SUMY = SUMY + Y(I)
      SUMX2 = SUMX2 + X(I)**2
    2 SUMXY = SUMXY + X(I)*Y(I)
      C = SUMX**2-N*SUMX2
      A = (SUMX*SUMY-N*SUMXY)/C
      B = (SUMX*SUMXY-SUMY*SUMX2)/C
      RETURN
      END
```

În lista de apel, N este numărul de date x şi y care se transmit subrutinei.

Pentru calculul valorilor lui k utilizăm ecuaţia 3.4.3. Singura dificultate care apare în această parte a programării este că funcţia ALØG din biblioteca calculatorului nu admite decît argumente mai mari decît 1. Pentru calculul lui k vom da organigrama unei funcţii-subrutină care evită dificultatea amintită mai sus:

Se dă în continuare lista de instrucţiuni a funcţiei denumite AK (vezi fig. 3.18)

```
      FUNCTIØN AK(AO,BO,X,T)
      MANT = 0
      Z = (1.-X*AO/BO)/(1.-X)
    1 IF(1.-Z)2,3,3
    3 Z = ALØG(Z)/2.303
      GØ TØ 5
    2 MANT = MANT-1
      Z = Z*10.0
      GØ TØ 1
    5 Z = Z + FLØAT(MANT)
      AK = 2.303*Z/(T*(BO-AO))
      RETURN
      END
```


Scanned with OKEN Scanner

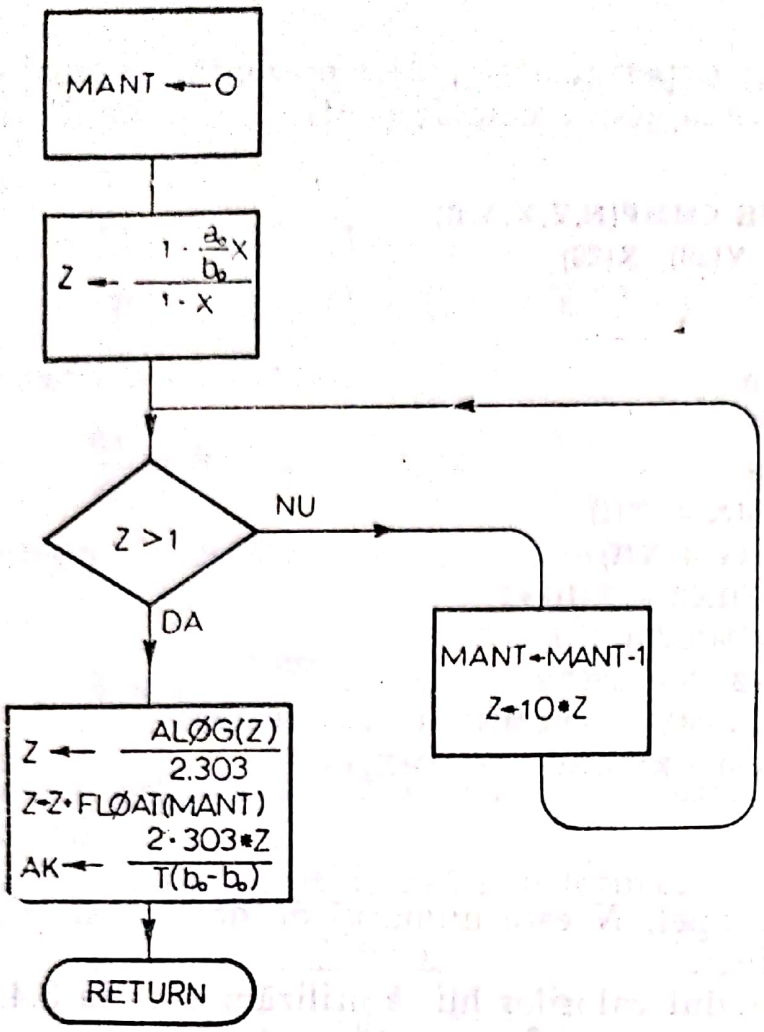


Fig. 3.18

Organigrama programului principal care calculează parametrii de activare pentru o reacţie de ordinul doi, utilizînd cele două subprograme prezentate mai înainte, este dată în fig. 3.19.

S-au folosit următoarele notaţii:

NT — numărul de temperaturi la care s-a lucrat

NJ — numărul de date la o temperatură

AR(I) — constantele de viteză medii pentru temperaturile I

X(J), T(J) — conversii şi timpi

TEMP(J) — temperaturile la care s-a lucrat

ARH — factorul Arrhenius

EACT — energia de activare

Valorile lui k calculate pentru fiecare X(I) vor fi stocate pînă la mediere chiar în X(I).

Programul are două variante de lucru:

NT = 1: nu se calculează ARH şi EACT ci doar AR;

NT ≠ 1: se calculează parametrii de activare

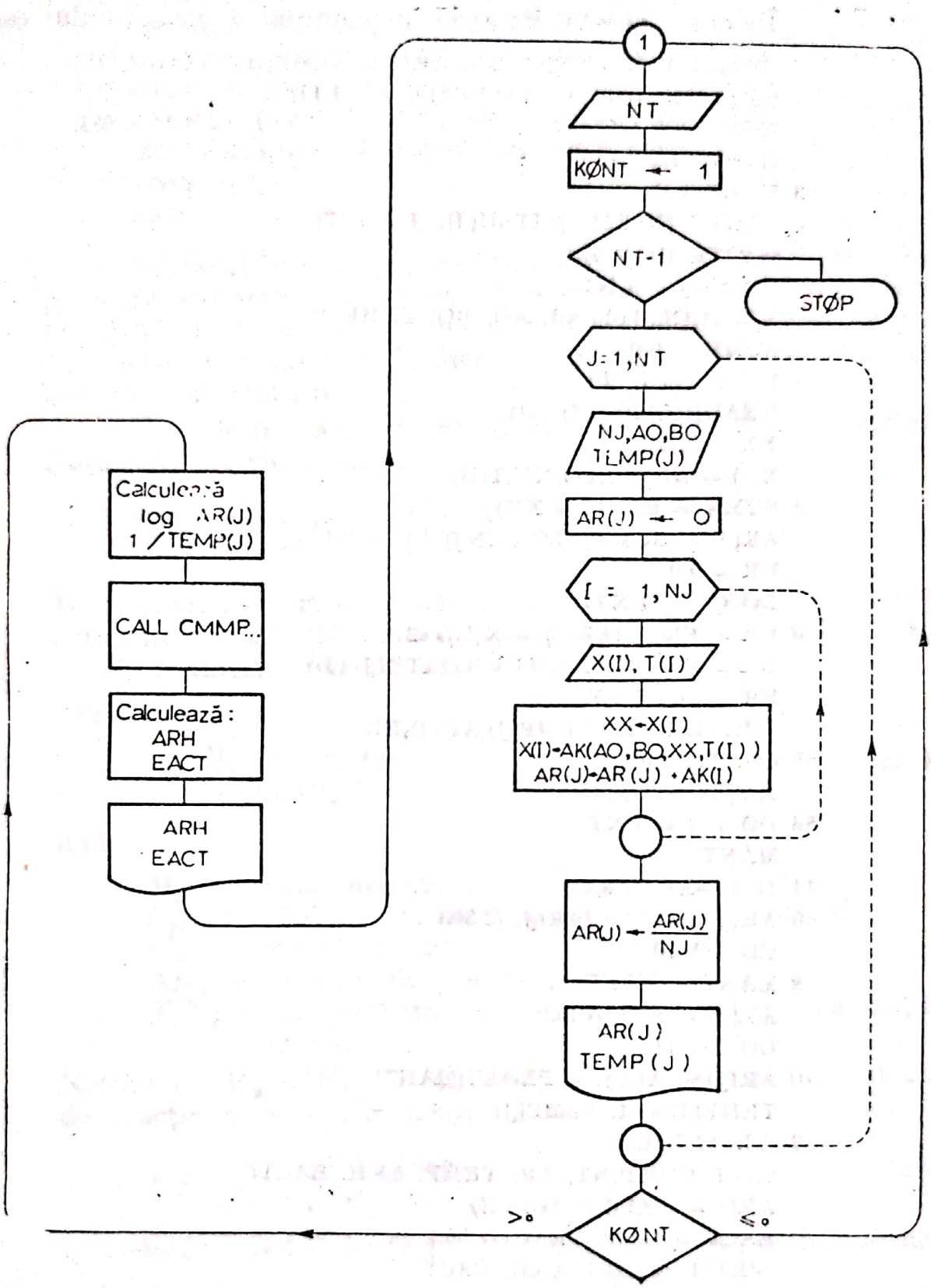


Fig. 3.19

Scanned with OKEN Scanner

În cazul acesta, lista de instrucțiuni a programului este:

```
      DIMENSIØN T(20), X(20), AR(10), TEMP(10), TITLU(17)
    1 READ(IN, 100)NT, (TITLU(J), J = 1,17)
      KØNT = 0
      IF(NT-1)2,3,4
    3 KØNT = 1
    4 WRITE (IØ,220) (TITLU(J), J = 1,17)
      WRITE (IØ,210)
      DØ 55 J = 1,NT
      READ(IN, 110) NJ, AO, BO, TEMP(J)
      SUMK = 0.0
      DØ 5 I = 1, NJ
      READ(IN,120)X(I),T(I)
      XX = X(I)
      X(I) = AK(AO,BO,XX,T(I))
    5 SUMK = SUMK + X(I)
      AR(J) = SUMK/FLØAT(NJ)
      ER = 0.0
      DØ 6 I = 1,NJ
    6 ER = ER + (AR(J) - X(I))**2
      ER = ER/(FLØAT(NJ)*FLØAT(NJ-1))
      ER = SQRT(ER)
      WRITE(IØ,200)TEMP(J),AR(J),ER
   55 CØNTINUE
      IF (KØNT)1,1,56
   56 DØ 7 J = 1,NT
      MÁNT = 0
   11 IF(1.-AR(J))8,9,9
    9 AR(J) = ALØG(AR(J))/2.303
      GØ TØ 10
    8 MANT = MANT-1
      AR(J) = AR(J)*10.0
      GØ TØ 11
   10 AR(J) = AR(J) + FLØAT(MANT)
      TEMP(J) = 1./TEMP(J)
    7 CØNTINUE
      CALL CMMP(NT, AR, TEMP, ARH, EACT)
      ARH = EXP(2.303*ARH)
      EACT = -1.986*EACT/2.303
      WRITE (IØ,230) ARH, EACT
      GØ TØ 1
  110 FØRMAT(I2,17A4)
  110 FØRMAT(I2,3F10.5)
```


Scanned with OKEN Scanner

```
120 FORMAT(2F10.5)
200 FORMAT(10X,F8.2,3X,E12.5,3X,E12.5,/)
210 FORMAT (10X, 'TEMPERATURA', 5X,' AK', 8X, 'ER. ST.',/)
220 FORMAT(1H1, 5X,17A4,//)
230 FORMAT (//,5X,'FACTOR ARRHENIUS', 8X,E12.5,/
    15X, 'ENERGIE DE ACTIVARE', 5X,E12.5)
  2 STOP
    END
```

În cazul în care experienţele sînt conduse adiabatic, temperatura sistemului se va modifica datorită căldurii de reacţie. De altfel, acest mod de lucru este interesant numai dacă căldura de reacţie este suficient de mare pentru a provoca variaţii sensibile ale temperaturii.

Considerînd acelaşi caz al reacţiei bimoleculare, variaţia conversiei în timp va fi

$$\frac{dx}{dt} = k\,[B]_0\,(1-x)\left(1-\frac{[A]_0}{[B]_0}\,x\right).$$

În continuare vom considera că reacţia se efectuează în soluţie suficient de diluată pentru a se putea considera căldura specifică a amestecului de reacţie constantă.

Atunci:

$$x = \frac{M_R \cdot C_p (T - T_0)}{N_{A_0} \Delta H_R}, \tag{3.4.6}$$

unde:

M_R — masa amestecului de reacţie (grame);
C_p — căldura specifică a amestecului (cal/g°C);
N_{A_0} — numărul iniţial de moli de A;
ΔH_R — căldura de reacţie (cal/mol);
T, T_0 — temperatura la un moment dat şi cea iniţială, în °K

Notînd că $M_R = V_R \cdot \bar{\rho}$, unde V_R — este volumul amestecului de reacţie şi $\bar{\rho}$ — densitatea medie, avem:

$$x = \frac{\bar{\rho} C_p (T - T_0)}{\Delta H_R (N_{A_0}/V_R)} = \frac{\bar{\rho} C_p\,(T - T_0)}{[A]_0 \cdot \Delta H_R}. \tag{3.4.7}$$

Introducînd (3.4.7) în (3.4.2) şi considerînd $A_0 = B_0$ obţinem:

$$\frac{dT}{dt} = k\,\frac{\Delta H_R}{\bar{\rho} C_p}\left(1 - \frac{\bar{\rho} \cdot C_p}{[A]_0\,\Delta H_R}\,(T - T_0)\right)^2. \tag{3.4.8}$$

Notînd $W = \bar{\rho} \cdot C_p / \Delta H_R$, trecînd la diferenţe finite şi substituind pentru k valoarea sa, obţinem din ecuaţia 3.4.8:

$$\frac{e^{\frac{E}{RT_i}}}{A} = \left(1 - \frac{W}{[A]_0}(T_i - T_0)\right)^2 / \left(W \cdot \left(\frac{\Delta T}{\Delta t}\right)_i\right).$$

Logaritmînd, obţinem:

$$\frac{2.303E}{RT_i} - \log A = \log\left(\left(1 - \frac{W}{[A]_0}(T_i - T_0)\right)^2 / \left(W \cdot \left(\frac{\Delta T}{\Delta t}\right)_i\right)\right) \quad (3.4.9)$$

Utilizînd din nou metoda celor mai mici pătrate, se pot calcula parametrii de activare E şi A din date care provin de la o singură experienţă.

Programul va trebui să aibă ca date de intrare:
T_0 — temperatura iniţială (°K)
$\bar{\rho}$ — densitatea medie a amestecului de reacţie (g/l)
C_p — căldura specifică medie (cal/g°K)
ΔH_R — căldura de reacţie (cal/mol)
A_0 — concentraţia iniţială a componentei A (mol/l)
— un tabel T_i, t_i

Pentru calculul expresiei de logaritmat din dreapta ecuaţiei 3.4.9 se va putea scrie o funcţie formulă, de exemplu:

```
F(A,B) = ((1. — RØ*CP*(A — TØ)/(AØ*DHR))**2)/
/(RØ*CP*B/DHR)
```

În instrucţiunea de chemare se va scrie de exemplu

```
TERM(I) = F(T(I), DT(I))
```

iar în program densitatea, căldura specifică, concentraţia iniţială şi căldura de reacţie vor trebui să apară cu numele RØ, CP, AØ şi respectiv DHR.

3.4.2. CAZURI COMPLEXE

În subcapitolul anterior am prezentat modul în care se pot determina constantele de viteză în cazul unor sisteme de reacţie compuse dintr-o singură reacţie, de ordin cunoscut. În acest capitol ne vom ocupa de sisteme de reacţie complexe, sisteme în cazul cărora calcularea constantelor de viteză ar fi greu de închipuit fără ajutorul calculatorului.

Scanned with OKEN Scanner

Considerăm ca exemplu un sistem de trei reacţii *paralel consecutive*:

$$A + B \xrightarrow{k_1} C + E$$

$$A + C \xrightarrow{k_2} D + E$$

$$A + D \xrightarrow{k_3} F + E$$

Acest sistem de reacţii se întîlneşte destul de des: de exemplu clorurările succesive ale unui substrat oarecare.

Un astfel de sistem este descris de următorul set de ecuaţii diferenţiale:

$$\frac{d[A]}{dt} = -k_1[A][B] - k_2[A][C] - k_3[A][D] \qquad (3.4.10)$$

$$\frac{d[B]}{dt} = -k_1[A][B] \qquad (3.4.11)$$

$$\frac{d[C]}{dt} = -k_2[A][C] + k_1[A][B] \qquad (3.4.12)$$

$$\frac{d[D]}{dt} = -k_3[A][D] + k_2[A][C] \qquad (3.4.13)$$

$$\frac{d[F]}{dt} = +k_3[A][D] \qquad (3.4.14)$$

Definim o nouă variabilă

$$\lambda = \int_0^t [A]\,dt \quad d\lambda = [A]\,dt \qquad (3.4.15-16)$$

Atunci ecuaţiile (3.4.10—14) devin

$$\frac{d[A]}{d\lambda} = -(k_1[B] + k_2[C] + k_3[D]) \qquad (3.4.17)$$

$$\frac{d[B]}{d\lambda} = -k_1[B] \qquad (3.4.18)$$

$$\frac{d[C]}{d\lambda} = k_1[B] - k_2[C] \qquad (3.4.19)$$

$$\frac{d[D]}{d\lambda} = k_2[C] - k_3[D] \qquad (3.4.20)$$

$$\frac{d[F]}{d\lambda} = k_3[D] \qquad (3.4.21)$$

Punînd condiţiile iniţiale

$$\lambda = C = D = 0,\ A = A_0,\ B = B_0$$

Scanned with OKEN Scanner

și efectuînd astfel experiențele încît $A_0 = 3B_0$, prin integrarea ecuațiilor de mai sus, obținem:

$$\frac{[A]}{[B]_0} = \left(3 + \frac{2k_1}{k_2 - k_1} + \frac{k_1 k_2}{(k_2 - k_1)(k_3 - k_1)}\right) e^{-k_1\lambda} - \left(\frac{2k_1}{k_2 - k_1} + \frac{k_1 k_2}{(k_2 - k_1)(k_3 - k_2)}\right) e^{-k_2\lambda} + \left(\frac{k_1 k_2}{(k_3 - k_1)(k_3 - k_2)}\right) e^{-k_3\lambda}. \quad (3.4.22)$$

Dintr-o astfel de ecuație sînt deosebit de greu de calculat constantele k_1, k_2 și k_3. Pentru asemenea cazuri se utilizează diverse metode de calcul prin aproximări succesive. Vom prezenta în continuare o astfel de metodă.

Ecuația 3.4.22 se poate rescrie în forma simplificată:

$$S = G_1 e^{-k_1\lambda} + G_2 e^{-k_2\lambda} + G_3 e^{-k_3\lambda}. \quad (3.4.23)$$

Dacă ${}^0k_1$, ${}^0k_2$, ${}^0k_3$ este un set de estimări inițiale ale constantelor de viteză reale, ecuația 3.4.23 poate fi dezvoltată în serie Taylor în jurul acestor valori:

$$S = (S)_0 + \left(\frac{\partial S}{\partial k_1}\right)_0 (k_1 - {}^0k_1) + \left(\frac{\partial S}{\partial k_2}\right)_0 (k_2 - {}^0k_2) + \left(\frac{\partial S}{\partial k_3}\right)_0 (k_3 - {}^0k_3) + H \quad (3.4.24)$$

Valorile $(S)_0$, $(\partial S/\partial k_1)_0$, $(\partial S/\partial k_2)_0$, $(\partial S/\partial k_3)_0$ se evaluează din ecuația 3.4.22 în care se înlocuiesc valorile lui ${}^0k_1$, ${}^0k_2$, ${}^0k_3$.

Neglijînd H și introducînd Δk_1, Δk_2, Δk_3,

$$S - (S)_0 = (\partial S/\partial k_1)_0 \Delta k_1 + (\partial S/\partial k_2)_0 \Delta k_2 + (\partial S/\partial k_3)_0 \Delta k_3 \quad (3.4.25)$$

Datele experimentale sînt redate sub forma unui tabel A funcție de timp. Din aceste date se calculează întîi valorile lui λ (integrînd grafic curba $A = f(t)$).

Ecuația (3.4.25) este de fapt de forma:

$$Y = X_1 \cdot \Delta k_1 + X_2 \cdot \Delta k_2 + X_3 \cdot \Delta k_3 \quad (3.4.26)$$

Avînd un set de valori pentru constantele de viteză, se pot calcula Y, X_1, X_2, X_3 pentru fiecare valoare a lui A și λ. Folosind metoda celor mai mici pătrate din ecuația 3.4.26 se obține următorul sistem (scris în formă matricială):

$$\begin{pmatrix} \sum X_1^2 & \sum X_1X_2 & \sum X_1X_3 \\ \sum X_2X_1 & \sum X_2^2 & \sum X_2X_3 \\ \sum X_3X_1 & \sum X_3X_2 & \sum X_3^2 \end{pmatrix} \times \begin{pmatrix} \Delta k_1 \\ \Delta k_2 \\ \Delta k_3 \end{pmatrix} = \begin{pmatrix} \sum YX_1 \\ \sum YX_2 \\ \sum YX_3 \end{pmatrix} \quad (3.4.27)$$

Scanned with OKEN Scanner

Rezolvînd sistemul se obţine un set de valori Δk_1, Δk_2, Δk_3 pentru corecţiile constantelor de viteză. Operînd corecţiile, noul set de constante se reintroduce în calcul. Ciclul se repetă pînă cînd valorile corecţiilor cad sub o anumită limită impusă iniţial.

Programul care va efectua aceste calcule va fi compus din :

— subrutină care calculează Y, X_1, X_2, X_3;
— subrutină care calculează sumele din sistemul 3.4.27;
— subrutină de rezolvat sisteme de ecuaţii liniare;
— program principal.

Schema logică simplificată a programului este :

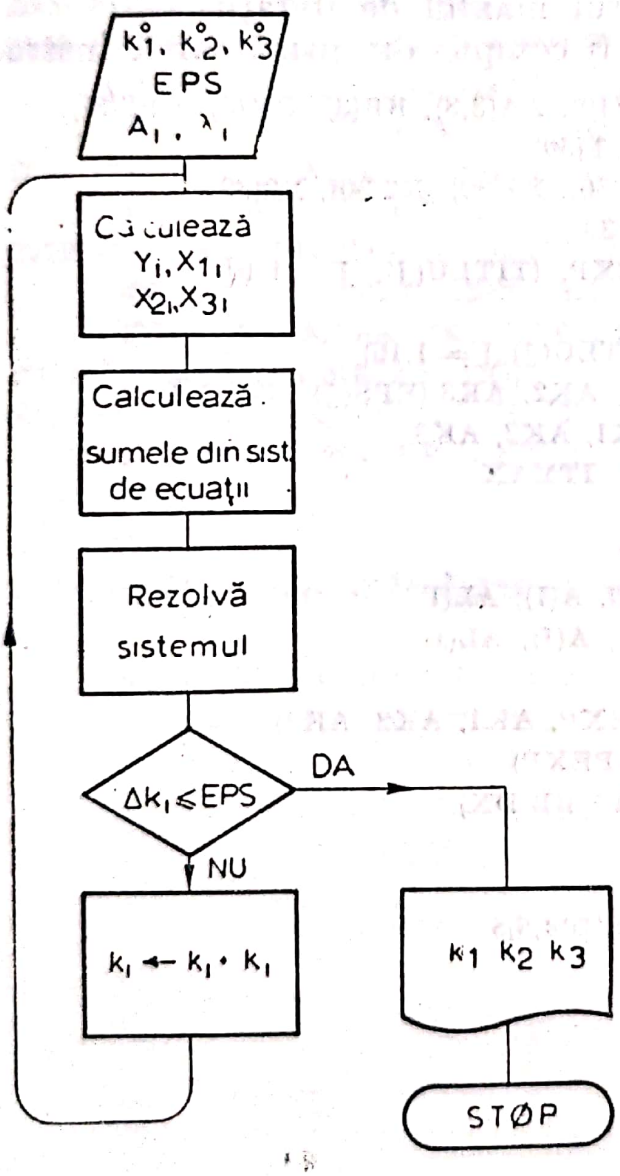


Fig. 3.20

La scrierea programului s-au folosit următoarele notaţii:
AA(I,J) — matricea coeficienţilor sistemului 3.4.27
BB(I) — termenii liberi ai sistemului 3.4.27
DX(I) — corecţiile constantelor
EPS(I) — precizia la fiecare din constante
T(J) — timp
BØ — concentraţia iniţială
A(J) — valorile concentraţiilor lui A
AL(J) — valorile lui λ
AK1, AK2, AK3 — constantele de viteză
X1(J), X2(J), X3(J) — X_1, X_2, X_3
S(J), SO(J) — Y din ecuaţia (3.4.26) şi $(S)_0$ din ecuaţia 3.4.25.
NPEXP — numărul de puncte experimentale
ITMAX — numărul maxim de iteraţii

Programul principal va fi compus din următoarele instrucţiuni:

```
      DIMENSIØN TITLU(10), AA(3,3), BB(3), DX(3), EPS(3),
     1                    T(30)
      CØMMØN A (30), AL(30), X1(30), X2(30), X3(30),
     1          ,S(30), SØ(30)
   99 READ (IN,1000)NPEXP, (TITLU(J), J = 1,10)
      IF (NPEXP)2,2,1
    1 WRITE (IØ,2000)(TITLU(J),J = 1,10)
      READ (IN,1010)AK1, AK2, AK3,(EPS(J), J = 1,3)
      WRITE (IØ, 2010)AK1, AK2, AK3
      READ(IN, 1020) BØ, ITMAX
      WRITE(IØ,2011)
      DØ 3 I = 1, NPEXP
      READ (IN, 1030) T(I), A(I), AL(I)
    3 WRITE (IØ,2030)T(I), A(I), AL(I)
      ITER = 1
   98 CALL CSX (BØ, NPEXP, AK1, AK2, AK3)
      CALL SUM(AA,BB,NPEXP)
      CALL GAUJØR(3,1,AA,BB,DX)
      IDC = 0
      DØ 4 I = 1,3
      IF (ABS(DX(I))-EPS(I))4,5,5
    4 CØNTINUE
      IDC = 1
    5 IF (IDC)6,6,8
    8 AK1 = AK1 + DX(1)
      AK2 = AK2 + DX(2)
      AK3 = AK3 + DX(3)
```


Scanned with OKEN Scanner

```
 97 WRITE(IØ,2020) ITER, AK1, AK2, AK3
    GØ TØ 99
  6 AK1 = AK1 + DX(1)
    AK2 = AK2 + DX(2)
    AK3 = AK3 + DX(3)
    ITER = ITER + 1
    IF(ITMAX-ITER)98,98,7
  7 WRITE (IØ,2040) ITMAX
    ITER = ITER-1
    GØ TØ 97
1000 FØRMAT(I2,10A4)
1010 FØRMAT (6E12.5)
1020 FØRMAT (F18.9,I2)
1030 FØRMAT(3F15.7)
2000 FØRMAT (1H1, 10X, 10A4)
2010 FØRMAT (//,10X,'ESTIMATIILE INITIALE PT.
   1 CØNSTANTELE 'DE VITEZA', /20X, 'K1 = ', E12.5,/20X,
   2 'K2 = ',E12.5, /20X,'K3 = ', E12.5///)
2011 FØRMAT (10X, 'DATELE EXPERIMENTALE SINT',//,
   1 10X,'T', 10X,' A', 10X, 'AL',/)
2020 FØRMAT (10X,'DUPA', I3,1X, 'ITERATII CONSTANTELE'
   1 'DE VITEZA SINT', /20X, 'K1 = ', E12.5,/20X,
   2 'K2 = ',E12.5,/20X, 'K3 = ',E12.5)
2030 FØRMAT (6X,3(E12.5, 1X))
2040 FØRMAT (10X, 'DUPA', I3, 1X,' ITERATII PRØCESUL
   1 'NU ESTE CØNVERGENT',/)
   2 STØP
     END
```

Subprogramul care calculează valorile Y, X_1, X_2, X_3 este următorul

```
      SUBRØUTINE CSX(BØ, NPEXP, B, C, D)
      CØMMØN A(30), AL(30), X1(30), X2(30), X3(30),
     1 S(30), SØ(30)
      G1 = 3. + 2.*B/(C-B) + B*C/((C-B)*(D-B))
      G2=-2.*B/(C-B)-B*C/((C-B)*(D-C))
      G3 = B*C/((D-B)*(D-C))
      G11 = (C*(D-B)**2 + C*D*(C + D-2.*B))/((C-B)**2*
     1       (D-B)**2)
      G21 = C*(C-2.*D)/((D-C)*((C-B)**2))
      G31 = C*D/((D-C)*((D-B)**2))
      G12 = -(B*(2.*D-B))/((D-B)*((C-B)**2))
      G22 = (B*D*(2.*D + B) - B*C*(4.*D-C))/(((C-B) **2)*
     1 ((D-C)**2))
```


Scanned with OKEN Scanner

```
      G32 = B*(B*C + C + D)/((D-B)*((D-C)**2))
      G13 = - B*C/((C-B) * ((D-B)**2))
      G23 = B*C/((C-B)*((D-C)**2))
      G33 = - B*C*(2.*D-B-C)/(((D-C)**2)*((D-B)**2))
      DØ 1 I = 1,NPEXP
      S(I) = A(I)/BØ
      A1 = EXP(-B*AL(I))
      A2 = EXP(-C*AL(I))
      A3 = EXP(-D*AL(I))
      SØ(I) = G1*A1 + G2*A2 + G3*A3
      S(I) = S(I) - SØ(I)
      X1(I) = (G11-G1*AL(I))*A1 + G21*A2 + G31*A3
      X2(I) = G12*A1 + (G22-G2*A1(I))*A2 + G32*A3
      X3(I) = G13*A1 + G23*A2 + (G33-G3*AL(I))*A3
    1 CØNTINUE
      RETURN
      END
```

Pentru calculul sumelor ce apar în sistemul 3.4.27 se foloseşte următorul subprogram :

```
      SUBRØUTINE SUM(AA,BB,NPEXP)
      DIMENSIØN AA(3,3), BB(3)
      CØMMØN A(30), AL(30), X1(30), X2(30), X3(30), S(30),
     1              ,SØ(30)
      DØ 5 J = 1,3
      BB(J) = 0.0
      DØ 5 K = 1,3
      AA(J,K) = 0.0
    5 CØNTINUE
      DØ 4 I = 1, NPEXP
      BB(1) = BB(1) + S(I)*X1(I)
      BB(2) = BB(2) + S(I)*X2(I)
      BB(3) = BB(3) + S(I)*X3(I)
      AA(1,1) = AA(1,1) + (X1(I)**2)
      AA(2,2) = AA(2,2) + (X2(I)**2)
      AA(3,3) = AA(3,3) + (X3(I)**2)
      AA(1,2) = AA(1,2) + X1(I)*X2(I)
      AA(1,3) = AA(1,3) + X1(I)*X3(I)
    4 AA(2,3) = AA(2,3) + X2(I)*X3(I)
      AA(2,1) = AA(1,2)
      AA(3,1) = AA(1,3)
      AA(3,2) = AA(2,3)
      RETURN
      END
```


Scanned with OKEN Scanner

BIBLIOGRAFIE

CAPITOLUL 1

1. C.P. POPOVICI, *Calculatoare cu program şi teoria programării*, Edit. Ştiinţifică, Bucureşti 1972.
2. P. DIMO, *Fortran IV*, Edit. Didactică şi Pedagogică, Bucureşti 1971.
3. N. COSTAKE, R. EFTIMIE, R. ZAMFIRESCU ŞI A. CHIRILĂ, *Fortran*, Edit. Tehnică 1971.
4. D.D. McCRACKEN, *A Guide to Fortran IV Programming*, Wiley, Inc., New York, N. Y. 1965.
5. M. DREYFUS, *Fortran IV*, Dunod, Paris 1969.

CAPITOLUL 2

1. A. GLÜCK, *Metode matematice în industria chimică*, Edit. Tehnică 1971.
2. J. A. N. LEE, *Numerical Analysis for Computers*, Reinhold Pbls. Corp. New York, N. Y. 1966.

CAPITOLUL 3

1. E. HÜCKEL, „Z. Physik", **70**, 20 (1931), ibid., **76**, 628 (1932).
2. R. B. WOODWARD ŞI R. HOFFMANN, „Angew. Chem.", **81**, 797 (1969) sau *The Conservation of Orbital Symmetry*, Verlag Chemie, Weinheim, 1970.
3. A. STREITWIESER, *Molecular Orbital Theory for Organic Chemists*, Wiley, New York, N. Y. 1961.
4. a. L. SALEM, *The Molecular Orbital Theory of Conjugated Systems*, W. A. Benjamin, Inc., New York, N. Y. 1966.
   b. R. G. PARR, *Quantum Theory of Molecular Electronic Structure*, W. A. Benjamin, Inc., New York, N. Y. 1963 (v. şi referinţa 15).
5. R. HOFFMANN, „J. Chem. Phys.", **39**, 1397, (1963), **40**, 2474, 2480, 2745 (1964).
6. Pentru metoda *CNDO* şi *INDO* v.: J. A. POPLE şi D. L. BEVERIDGE, *Approximate Molecular Orbital Theory*, McGraw-Hill, New York, N. Y. 1970.

Scanned with OKEN Scanner

29. v. referința 6, pag. 81.

30. P. DIEHL, E. FLUCK ȘI R. KOSFELD, *NMR-Basic Principles and Progress*, vol. 6. — *Computer Assistance in the Analysis of High-Resolution NMR Spectra*, Springer Verlag, Berlin, 1972.

31. S. CASTELLANO și A. A. BOTHNER-BY, „J. Chem. Phys.", **41**, 3863 (1964).

32. J.D. SWALEN ȘI C.A. REILLY, ibid., **37**, 21 (1962).

33. DeLOS F. DeTAR Ed., *Computer Programs for Chemistry*, W. A. Benjamin, Inc., 1968, vol 1, Cap. 3 și Cap. 4.

Scanned with OKEN Scanner

GLØSAR

Termeni folosiţi curent în lucrări privind utilizarea calculatoarelor. Deoarece majoritatea textelor sursă sînt în limba engleză, din al cărei vocabular s-au preluat şi termenii folosiţi în FØRTRAN, alături de terminologia românească, în anumite cazuri (în care există o diferenţă între termenul românesc şi cel englezesc) este indicată şi terminologia originală.

ACCES — timpul necesar condiţionării unui dispozitiv de memorare de a primi sau transmite date, după ce comanda de a face acest lucru a fost executată.

ACUMULATØR

1. Registrul şi echipamentul aferent din unitatea aritmetică, în care se execută operaţiile aritmetice şi logice.
2. Unitate a calculatorului numeric unde sînt totalizate numerele, adică acumulate.

ACURATEŢE — Gradul de exactitate al unei aproximaţii sau măsurători. În mod normal acurateţea denotă calitatea absolută a rezultatelor în timp ce precizia, în mod obişnuit, se referă la cantitatea de detaliu folosită în reprezentarea acelor rezultate.

ADRESĂ

1. O identificare a unui registru sau a unei alocaţii din memorie, reprezentată de un nume, etichetă sau număr.
2. Acea parte dintr-o instrucţiune care specifică operandul acelei instrucţiuni.

ALFANUMERIC — Caractere care cuprind: literele alfabetului (A—Z), numerele de la 0 la 9 şi simboluri speciale ca de exemplu —,/,*,.,(,), +, etc.

ALGØRITM — Un set de reguli sau procedee bine stabilite care servesc la soluţionarea unei probleme într-un număr finit de etape.

ALGØL — *ALGØrithmic Language* — un limbaj orientat pentru rezolvarea unor probleme ştiinţifice.

ALØCARE — Procesul de rezervare a unor zone din memorie pentru stocarea anumitor informaţii.

Scanned with OKEN Scanner

ANALOG — Reprezentarea cantităţilor numerice cu ajutorul unor variabile fizice, ca de exemplu: translaţie, rotaţie, voltaj, rezistenţe, etc.

ASSEMBLER — Un program care operează asupra datelor simbolice de intrare (programe scrise în limbaje diverse, date numerice, etc.) pentru a îndeplinii funcţii ca:

- traducerea codului de operaţii simbolice în instrucţiuni de operare ale maşinii
- atribuirea de adrese în memorie unor instrucţiuni succesive

BASIC — *Beginner's All-purpose Symbolic Instruction Code* este o versiune simplificată a limbajului FORTRAN

BCD — Abrevierea pentru *Binary Coded Decimal*

BIBLIOTECA — (Library) — colecţie de subprograme standard, bine testate şi care sînt depozitate pe bandă sau discuri magnetice.

BINAR — O caracteristică, proprietate sau condiţie în care nu există decît două alternative.

BIT — Un caracter dintr-un număr binar (0 sau 1)

BUGS — Nu are echivalent în limba română ca semnificaţie în acest domeniu. Desemnează erori de programare nedetectabile de calculator.

BYT — O secvenţă de biţi adiacenţi asupra cărora se operează ca asupra unei unităţi. De obicei este mai scurt decît un cuvînt. Unul din cele mai răspîndite grupări este cea de 8 biţi (IBM 360)

CARACTER ILEGAL — Un caracter sau combinaţie de biţi care nu este acceptată de către calculator ca fiind validă sau ca fiind o reprezentare cunoscută.

CELULA — Memoria necesară stocării unei unităţi de informaţie.

CICLU DE EXECUŢIE — Timpul în care o instrucţiune a maşinii este interpretată şi operaţia indicată este executată asupra operandului specificat.

COBOL — *COmmon Business Oriented Language* — limbaj orientat pentru rezolvarea problemelor economice.

COD — Atribuirea unui anumit interes unui grup de simboluri, numere sau caractere.

COMPILARE — Producerea unui limbaj intermediar sau de maşină dintr-o rutină scrisă într-un limbaj sursă evoluat.

Scanned with OKEN Scanner

Această traducere se realizează de către un sistem de programe denumit CØMPILATOR.

CUVÎNT — Un set ordonat de caractere care ocupă o celulă de memorie şi care sînt tratate de circuitele calculatorului ca o unitate.

DEBUG — Localizarea şi corectarea tuturor greşelilor din program. Poate să mai însemne şi detectarea şi corectarea unor proaste funcţionări ale calculatorului sau perifericelor sale. Se efectuează cu programe depanatoare ca:

- *TRACE* — analiza fiecărei instrucţiuni executate şi scrierea ei pe un dispozitiv de ieşire
- *SNAPSHØT DUMP* — o tipărire a rezultatelor calculului, în punctele de întrerupere sau de verificare, aflate în anumite zone ale memoriei.
- *PØST MØRTEM DUMP* — acelaşi lucru ca mai sus însă numai cînd programul se opreşte

DELETE — A îndepărta sau a elimina.

DUBLĂ PRECIZIE — Rezervarea pentru anumite mărimi a unui număr dublu de celule de memorie faţă de modul de rezervare obişnuit.

EBCDIC — *Extended Binary Coded Decimal Interchange Code*-codul care foloseşte 8 biţi (vezi IBM 360).

ECHIPAMENTE — Părţi componente ale calculatorului. Există următoarele tipuri:

- de intrare (INPUT):
  cititor de cartele
  maşină de scris
  cititor de bandă perforată
  cititor optic
  unităţi de bandă magnetică
- de ieşire (ØUTPUT)
  imprimantă rapidă
  înregistrator
  perforator de cartele
  unităţi de bandă magnetică
- periferice — instalaţii auxiliare care pot fi plasate sub controlul unităţii centrale a calculatorului: de exemplu toate echipamentele I — E.

EDITARE — Rearanjarea datelor sau informaţiilor.

Scanned with OKEN Scanner

EROARE — Termenul general se referă la orice deviere a unei cantități calculate sau măsurate de la valoarea ei teoretică corectă sau de la valoarea ei adevărată.

HARDWARE — Părțile magnetice, electrice, electronice și mecanice din care este construit calculatorul.

IOCS — *Input-Output Control Sistem*

INSTRUCȚIUNE — Un set de caractere care definesc împreună cu una sau mai multe adrese o operație și operanzi corespunzători.

INTERFAȚĂ — Zona de contact dintre două piese de hardware sau dintre două sisteme.

LIMBAJ — Un vocabular însoțit de reguli de sintaxă.
Există limbaj mașină (limbaj folosit direct de mașină, care nu necesită traducere), limbaj obiect (limbaj produs de un sistem de codificare automat) și limbaj sursă (limbajul original al programului).

LINKAGE — Legătură; o tehnică de întoarcere la un program de la o subrutină închisă. Subrutina închisă este orice subprogram care nu este inclus în programul principal.

LOCAȚIE — Poziție de stocare a unei instrucțiuni în memoria internă.

LUNGIME — Numărul de caractere, biți sau byts dintr-un cuvînt.

MEMORIE — O parte a calculatorului unde se poate depozita reversibil informație. Poate să existe:
memorie internă — face parte din structura de bază a calculatorulni
memorie auxiliară (externă) — unități de bandă, disc, tambur magnetic.

PROCES — Termen generic care poate implica calculare, compilare, asamblare, etc.

PROGRAM SUPERVIZOR — Un sistem de programe care controlează funcționarea tuturor programelor din mașină; operațiile I—E, compilatorul, inițierea și execuția programelor, etc.

PL—1 — Limbaj de programare cu aplicabilitate foarte largă, care îmbină calitățile limbajelor FORTRAN, ALGOL și COBOL.

SEGMENTARE — Împărțirea unui program în părți compuse din cîte un număr întreg de subprograme.
Fiecare segment în parte poate fi depozitat în memoria centrală.

Scanned with OKEN Scanner

SISTEM DE ØPERARE	Este similar cu program supervizor.
SØFTWARE	Colecţia de programe necesare funcţionării unui calculator.
UNITATE CENTRALĂ	Partea dintr-un calculator care conţine circuitele aritmetice, logice şi de control şi interfeţele pentru I—E.
ZØNA TAMPØN	Dispozitiv folosit pentru a păstra temporar informaţiile care se transmit între dispozitive ale memoriei interne şi externe sau între acestea şi dispozitivele I—E.

REDACTOR: DELIA IACOB
TEHNOREDACTOR: FLORICA WEIDLE

COLI DE TIPAR: 16,75.
TIRAJUL: 2800.

ÎNTREPRINDEREA POLIGRAFICĂ CLUJ
STR. BRASSAI NR. 5—7
REPUBLICA SOCIALISTĂ ROMÂNIA
COMANDA NR. 204/73

Scanned with OKEN Scanner